Partitioned Fluid-Structure Interaction for Full Rotor Computations Using CFD

Heinz, Joachim Christian

Publication date:
2013

Document Version
Publisher's PDF, also known as Version of record

Partitioned Fluid-Structure Interaction for Full Rotor Simulations Using CFD

DTU Wind Energy - PhD

Joachim Christian Heinz
DTU Wind Energy PhD-0033(EN)
June 2013

DTU Wind Energy
Department of Wind Energy

Author: Joachim Christian Heinz

Title: Partitioned Fluid-Structure Interaction for Full Rotor Computations Using CFD

Department: DTU Wind Energy (Campus Risø)

This dissertation is submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy in Engineering at the Technical University of Denmark. It is based on the work done during a three years Ph.D. study at the Aeroelastic Design Section of the Department of Wind Energy at Risø Campus. The work was partly funded by the ATEF project on adaptive trailing edge flaps for wind turbines sponsored by Højteknologifonden. The dissertation was submitted in February 2013 and successfully defended the 17th of June 2013.

Main Supervisor: Research Professor, Niels N. Sørensen, DTU Wind Energy.
Co-Supervisor: Senior Scientist, Frederik Zahle, DTU Wind Energy.
Co-Supervisor: Senior Researcher, John M. Hansen, DTU Wind Energy.

DTU Wind Energy PhD-0033(EN)
June 2013

ISBN:
978-87-92896-74-2

Sponsorship:
Højteknologifonden, Advanced Technology Projects 2007, Development of ATEF system for wind turbines

Pages: 162
Figures: 72
References: 76

Technical University of Denmark
Department of Wind Energy
Frederiksborgvej 399
Building 118
4000 Roskilde
Denmark
jhei@dtu.dk
www.vindenergi.dtu.dk

Partitioned Fluid-Structure Interaction for Full Rotor Computations Using CFD

Ph.D. Thesis

by

Joachim Christian Heinz

June 2013

Technical University of Denmark
Department of Wind Energy
Aeroelastic Design
DTU Risø Campus

Abstract

In the design of modern wind turbines with long and slender rotor blades it becomes increasingly important to model and understand the evolving aero-elastic effects in more details. Standard state-of-the-art aero-elastic simulation tools for wind turbines usually employ a blade element momentum (BEM) based aerodynamic model which is computationally cheap but includes several limitations and corrections in order to account for three-dimensional and unsteady effects. The present work discusses the development of an aero-elastic simulation tool where high-fidelity computational fluid dynamics (CFD) is used to model the aerodynamics of the flexible wind turbine rotor. Respective CFD computations are computationally expensive but do not show the limitations of the BEM-based models. It is one of the first times that high-fidelity fluid-structure interaction (FSI) simulations are used to model the aero-elastic response of an entire wind turbine rotor.

The work employs a partitioned FSI coupling between the multi-body-based structural model of the aero-elastic solver HAWC2 and the finite volume CFD solver EllipSys3D. In order to establish an FSI coupling of sufficient time accuracy and sufficient numerical stability several coupling strategies are investigated and implemented. The considered coupling strategies incorporate both loose and strong coupling schemes and employ both a conservative and a non-conservative force and deflection transfer. In a specific assessment of the implemented coupling schemes it was found that a relatively simple loosely coupled algorithm with a non-conservative force transfer is well-suited to establish a second order time accurate and sufficiently stable FSI simulation. The use of a strong coupling scheme was found to be redundant.

Results of the partitioned FSI coupling between HAWC2 and EllipSys3D (HAWC2CFD) were then compared to the computations of the stand-alone solver of HAWC2 which employs traditional BEM theory to model the aerodynamics. In a first set of comparative simulations the quasi-steady aero-servo-elastic response of the NREL 5MW reference wind turbine was investigated for the wind

speed range between 4 m/s and 24 m/s. In a second test case the same turbine was modelled during an emergency shut-down due to a loss of power in which the rotor blades are quickly pitched to feather in order to slow down the turbine. The rapid change in the aerodynamic loading and the severe structural response evoke complex flow regimes which are rather challenging to model with the traditional BEM-based models. The comparisons between the results of HAWC2CFD and HAWC2 revealed a very good agreement in the predicted aero-servo-elastic response of the modelled wind turbine, although some smaller discrepancies could be found in the predicted aerodynamic forces.

Additionally, the work includes the description of a generic coupling framework which was developed in order to establish the desired partitioned coupling between HAWC2 and EllipSys3D. The developed framework was then used to also conduct FSI simulations of isolated two-dimensional and three-dimensional aerofoil sections by coupling a simple three degrees of freedom structural model with the respective CFD model.

Resumé

Designs af moderne mega-watt vindmøller benytter sig i stigende grad af slanke elastiske vinger, hvilket øger kravene til både simuleringsmodeller samt forståelse af de komplekse aeroelastiske effekter, der er på spil. De aerodynamiske modeller i aeroelastiske værktøjer til vindmøller er normalt baseret på *blade element momentum* (BEM) teori, som er beregningsmæssigt hurtige, men har adskillige begrænsninger, deriblandt at de for eksempel bygger på korrektioner for tredimensionelle og instationære effekter. Dette arbejde omhandler udviklingen af et aeroelastisk simuleringsværktøj, hvor aerodynamikken for den fleksible vindmøllerotor er baseret på *computational fluid dynamics* (CFD). Denne type modellering har ikke de samme begrænsninger som BEM-baserede modeller, men er til gengæld beregningsmæssigt væsentligt dyrere. Dette arbejde præsenterer således, som en af de første, aero-elastiske simuleringer af en komplet vindmøllerotor, med brug af avancerede *fluid-structure interaction* (FSI) simuleringer.

I dette arbejde bruges en såkaldt partitioneret FSI kobling mellem multi-body strukturmodellen i den aeroelastiske løser HAWC2 og finite-volume CFD løseren EllipSys3D. For at opnå tilstrækkelig tidsnøjagtighed samt numerisk stabilitet blev adskillige koblingsstrategier implementeret. Disse omfattede både løse og stærke koblingsstrategier samt brug af både konservativ og ikke-konservativ overførsel af krafter og deformationer. Den detaljerede evaluering af de implementerede koblingsstrategier viste, at den relativt simple løst koblede algoritme med ikke-konservativ overførsel af kræfter var tilstrækkelig for at opnå anden ordens tidsnøjagtighed samt numerisk stabilitet af FSI simuleringerne. Det blev konkluderet at en stærk kobling mellem løserne ikke var nødvendigt.

Simuleringer foretaget med den koblede HAWC2 og EllipSys3D løser (HAWC2CFD) blev derefter sammenlignet med standardversionen af HAWC2 som bruger BEM til at modellere aerodynamikken. I den første serie af simuleringer blev de kvasi-statiske aero-servo-elastiske egenskaber af NREL 5 MW vindmøllen sammenlignet for vindhastigheder mellem 4 m/s og 24 m/s. Den følgende undersøgelse blev foretaget på den samme mølle for et lasttilfælde med

nødstop af møllen som følge af strømafbrydelse, hvor vingerne hurtigt vinkles ind i vinden for at bremse rotationen. Som følge af de pludselige ændringer i den aerodynamiske last og de resulterende store strukturelle udbøjninger er dette last-tilfælde særdeles komplekst at simulere med BEM-baserede modeller. Sammenligningerne mellem HAWC2CFD og HAWC2 viste god overensstemmelse mellem de to koder i simuleringen af de aero-servo-elastiske egensskaber for vindmøllen på trods af mindre uoverensstemmelser i forudsigelsen af de aerodynamiske kræfter. Dette arbejde præsenterer yderligere beskrivelsen af det generiske koblingskompleks som blev udviklet til at etablere den partitionerede kobling mellem HAWC2 og EllipSys3D. Dette kompleks blev også brugt til FSI simuleringer af isolerede to- og tredimensionelle vingesektioner med kobling mellem en simpel strukturmodel med tre frihedsgrader og den tilsvarende CFD løser.

Acknowledgements

I would like to thank Niels N. Sørensen and Frederik Zahle for their competent and devoted supervision. I am thankful for the numerous ideas and inspirations they gave me and for all the time they found in order to provide me the help and advice I needed. I especially appreciate the confidence they had in me to give me the freedom of developing and conducting this Ph.D. project in my own preferred way. I would also like to thank my third supervisor John M. Hansen. The many hours he spent in order to mitigate the aero-elastic code HAWC2 from Windows to Linux were a crucial effort for the realization of my project.

Apart from my three supervisors I would also like to thank my co-workers at the Aeroelastic Design Section of DTU Wind Energy. In particular, I would like to thank Anders M. Hansen who provided me a lot of information and help with respect to the structural modelling of HAWC2. I would also like to thank Mads Kristensen and Brian Vinter from the E-Science Center at the University of Copenhagen for supporting me at the very early stage of my work.

Within those last three years of research I was allowed to experience a lot of sympathy, support and love from people that are important for me. I deeply appreciate all the memories connected to them, they make me feel rich and happy. In particular I want to mention my brother Alexander and my parents Richard and Eveline who were and are always there for me.

Preface

This dissertation is submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy in Engineering at the Technical University of Denmark. It is based on the work done during a three years Ph.D. study at the Aeroelastic Design Section of the Department of Wind Energy at Risø Campus. The work was partly funded by the ATEF project on adaptive trailing edge flaps for wind turbines sponsored by Højteknologifonden.

Nomenclature

$[\,]^*$	Superscript indicating Quantities of Predicted Deflection State
$[\,]^n$	Superscript indicating Quantities of Current Time Step
$[\,]^{n+1}$	Superscript indicating Quantities of New Time Step
$[\,]_\Gamma$	Subscript indicating Quantity of Wet Boundary Surface
$[\,]_F$	Subscript indicating Quantities of Fluid Solver
$[\,]_S$	Subscript indicating Quantities of Structure Solver
$[\,]_{def}$	Subscript indicating Effective Deflection (Total minus Initial Deflection)
$[\,]_{ini}$	Subscript indicating Initial Deformation State
$\bar{[\,]}$	Time-Averaged Quantity
$\tilde{[\,]}$	Non-Scaled Quantity
2D	Two Dimensions
3-DOF	Three Degrees of Freedom
3D	Three Dimensions
α_c	Cone Angle
α_t	Tilt Angle
α_y	Yaw Angle
$\bar{\kappa}_{ij}$	Time-Averaged Volume Change due to Motion of Cell Face ij
$\bar{\boldsymbol{\mu}}_{ij}$	Time-Averaged Geometrical Quantity for ALE Convective Fluxes

$\boldsymbol{\eta}$	Set of Interpolation Function
$\Delta E_{F,FS}$	Energy released by Fluid Solver during Force Transfer from Fluid To Structure
$\Delta E_{F,SF}$	Energy of Fluid Solver during Deflection Transfer from Structure to Fluid
ΔE_{Loose}	Inherent Energy Error of a Loose Coupling Scheme
$\Delta E_{S,FS}$	Energy received by Structure Solver due to Force Transfer from Fluid to Structure
$\Delta E_{S,SF}$	Energy of Structure Solver during Deflection Transfer from Structure to Fluid
ϵ	Twist Angle
Γ	Wet Boundary Surface
Φ	Azimuthal Angle
ρ_F	Density of Fluid
ρ_S	Density of Structure
Θ	Pitch Angle
$\bar{\mathbf{x}}_g$	Time-Averaged Mesh Position for ALE Diffusive Fluxes
$\mathbf{F}$	Blend Factors for CFD Mesh Deformation
$\mathbf{F}_F$	Aerodynamic Forces in Fluid Mesh Discretization
$\mathbf{F}_S$	Aerodynamic Forces in Structure Mesh Discretization
$\mathbf{N}$	Set of Extrapolation Function
ub	Structure Mesh Positions of Blades (Exclusive Hub)
uh	Structure Mesh Positions of Hub (Exclusive Blades)
$\mathbf{U}$	Mesh Deformation Matrix
u	Structure Mesh Positions

xb	Fluid Mesh Positions due to Blade Deformation (Exclusive Hub Motion)
xh	Fluid Mesh Positions due to Hub Motion (Exclusive Blade Deformation)
x	Fluid Mesh Positions
$\mathbf{x}_u$	Fluid Mesh Positions projected on Structure Mesh
D	Sectional Drag Force
L	Sectional Lift Force
M	Sectional Moment
r/R	Relative Blade Position
t	Time
U_∞	Inflow Velocity at Farfield
F	Fluid Solver
S	Structure Solver
ALE	Arbitrary Lagrangian-Eulerian
BEM Theory	Blade Element Momentum Theory
CFD	Computational Fluid Dynamics
CSS	Conventional Serial Staggered Grid
DTU	Technical University of Denmark
EL2D	Four Letter Code for EllipSys2D
EL3D	Four Letter Code for EllipSys3D
EllipSys2D	Two-Dimensional Finite Volume RANS Solver
EllipSys3D	Three-Dimensional Finite Volume RANS Solver
EUL	Eulerian
FCTR	Four Letter Code for Flap Control Model

FEM	Finite Element Method
FS Interface	Fluid-Structure Interface
FSI	Fluid-Structure Interaction
GCL	Geometrical Conservation Law
GSS	Generalized Serial Staggered Grid
HAWC	Four Letter Code for HAWC2
HAWC2	Aero-Elastic Code for Horizontal Axis Wind Turbines
HAWC2CFD	High-Fidelity FSI coupling between HAWC2 and EllipSys3D
MPI	Message Passing Interface
QUICK	Quadratic Upstream Interpolation for Convective Kinematics
RANS	Reynolds-Averaged Navier-Stokes Equations
STRC	Four Letter Code for 3-DOF Structural Model
SUDS	Second-Order Accurate Upwind Difference Scheme
BLCS	Blade Coordinate System
BSCS	Blade Section Coordinate System
GLCS	Global Coordinate System
HUCS	Hub Coordinate System
SHCS	Shaft Coordinate System

Contents

Chapter 1

Introduction

1.1 General Motivation

In recent years the importance of creating a sustainable and thus environmental friendly way of living got recognised in many societies. This increased awareness of sustainability meanwhile also influences the governmental decisions in many countries and lead to several dedicated investments in the energy sector in order to establish a sustainable energy production. Until the year 2020 the European Union has the objective to increase the amount of renewable energies to a share of 20 % of the total energy consumption. Denmark wants to contribute to this commitment by increasing their share to 30 % while Germany wants to elevate their share to 35 % in the same time span. A long term commitment for the year 2050 states that Germany wants to increase their share of renewable energies to 80 % of the total energy consumption. In the United States of America recent initiatives define the objective that 25 % of the US energy needs should be covered with renewable energies by 2025. In all respective scenarios wind energy plays a major role in order to attain this decisive expansion of renewable energies.

In order to make wind energy cheaper and even more competitive in the future, the recent research activities place the focus on building larger wind turbines. In this way the ratio between turbine cost and produced energy can be further reduced. However, a simple up-scaling of existing wind turbine models would result in inappropriately designed and heavy structural components and would put unreasonably high loads on the rotor bearings and the entire support structure. In order to reduce the weight of those components relative to the turbine size several key components such as the wind turbine blades have to be redesigned

from scratch. The blades of future multi-megawatt wind turbines will be relatively light but will thus also be relatively slender and flexible. The development of wind turbines with such flexible structural components demands for a better understanding of the increased aero-elastic coupling between the structural deformation and the corresponding aerodynamic forces in order to eliminate undesired vibrations or destructive flutter. Appropriate simulation tools have to be used to model and understand those aero-elastic effects and to develop strategies which can mitigate and control them.

Several simulation tools of different complexity exist in order to investigate the aero-elastic interaction between the aerodynamic loads and the structural response of a wind turbine. However, within the present work a high-fidelity fluid-structure interaction (FSI) simulation tool is developed that constitutes another level of aero-elastic modelling. It couples the multi-body based structural model of the aero-elastic code HAWC2 [1] with the three-dimensional full rotor computations of the computational fluid dynamics (CFD) code EllipSys3D [2], [3], [4]. Employing high-fidelity CFD computations within an FSI simulation permits numerous new insights to the aero-elastic effects that occur on modern wind turbines. It also provides the opportunity to assess and validate the traditional and well established simulation tools of lower complexity and of severely reduced computational costs.

1.2 Fluid-Structure Interaction

Many technically relevant problem statements involve the interaction between different physical domains. Traditionally, those multifield problems had to be split up into less complex and manageable one field problems where the absent fields were either ignored or modelled via heavily simplified boundary conditions. With an increased understanding and an increased quality of the developed one field models it became desirable to also model and understand the influences of the other fields in more detail. In recent years the considerable increase in computational power finally paved the way for high-fidelity multifield simulations which employ detailed formulations of all involved fields and provide a detailed modelling of the existing inter-field coupling. Respective multi-field problems can be found in the modelling of e.g. the electro-mechanical interaction in the cardiac muscle, the electro-magnetic interferences in power electronics, the thermo-structural dependencies in order to determine thermal stresses and the influences

of chemical reactions to fluid flows. Further and more sophisticated combinations and interactions of the above mentioned multifield problems are obvious.

A multifield problem of major interest is the interaction between fluids and structures. Within the scope of Fluid-Structure Interaction (FSI) analysis a considerable amount of research activities can be found in the fields of aeronautics [5], [6], [7], [8] and biomechanics [9], [10], [11], [12] but typical examples also include the investigation of offshore structures subjected to waves [13], bridges subjected to wind [14], the performance of parachutes [15] or airbags [16] and the behaviour of liquid storage tanks during earthquakes and transportation [17]. An increased interest in FSI simulations can be recently found in the field of wind energy and a comprehensive introduction into the respective activities is given in Section 1.2.2.

1.2.1 Partitioned vs. Monolithic Coupling

The simulation of multifield problems in general and FSI problems in particular involves the modelling of different physical domains. In a partitioned coupling approach each of the considered physical domains is formulated in an independent subsystem and the interaction is achieved by exchanging the necessary data through a common interface. In contrast to this a monolithic coupling approach incorporates all physical effects into one common system of equations which can then be treated as one single entity.

The partitioned approach has several advantages when compared to the monolithic approach. Keeping the involved subsystems as independent entities allows to model the multifield problem using efficient and well-tested stand-alone solvers where each solver is specialized for its particular scope of application, employing well-suited discretization and time integration techniques. A partitioned coupling system is very flexible and the achieved software modularity makes it relatively painless to exchange the participating solvers. The modularity also reduces the complexity of the overall system and makes it easier to access and further optimize the involved subsystems when better mathematical models and methods emerge. However, there are also some drawbacks involved when partitioned coupling approaches are employed, and depending on the implementation and considered cases severe stability and accuracy problems can occur [10].

A monolithic coupling is especially developed for one particular coupling problem. The resulting system of equations is rather complex and inflexible and applications are mostly limited to simplified academic problems. In an academic context it can be advantageous to formulate the considered multifield problem with one single system of equations in order to allow a detailed mathematical analysis. In most other cases it is usually not beneficial to employ a monolithic coupling approach since the earlier reported stability and accuracy problems of the partitioned coupling approaches can be controlled if suitable coupling schemes are employed.

1.2.2 Full Rotor FSI in Wind Energy

The development of modern wind turbines with structural components of decreased weight and increased flexibility require a good understanding of the coupling effects between the deforming structure and the corresponding aerodynamics. This fluid-structure or aero-elastic interaction can be modelled in different levels of complexity and a comprehensive overview of the employed methods within wind energy can be found in Hansen [18] and Zhang [19].

The aero-elastic response of an entire wind turbine is typically simulated with popular time-domain codes like FAST [20], FLEX5 [21], BLADED [22], PHATAS [23], GAST [24] and the above mentioned HAWC2 [1]. The structural models of those codes are either based on mode shape functions, linear or non-linear beam elements or on a multi-body approach using one-dimensional beams. The respective aerodynamic models are usually based on the blade element momentum (BEM) theory, some few aero-elastic codes like PHATAS and GAST are also capable of employing vortex methods. Computationally expensive but more accurate models like high-fidelity finite element methods (FEM) for the structure or high-fidelity computational fluid dynamics (CFD) for the aerodynamics are also heavily integrated in the aero-elastic design process. However, these methods are usually employed in stand-alone investigations without a direct coupling to a complementary aerodynamic or structural solver and have not yet been taken into serious consideration in order to simulate the aero-elastic response of a wind turbine.

Detailed stand-alone FEM simulations are commonly used in wind energy research in order to model the structural deformations of an isolated wind turbine

blade and to give valuable information about the local stresses within the composite materials. However, so far it did not seem reasonable to employ the computationally heavy FEM calculations in aero-elastic simulations of entire wind turbines. In this context it might be interesting to refer to the work of Kallesøe [25] where a detailed FEM model of a 34 m long wind turbine blade is used to determine the resulting cross-sectional deformations of a deflected blade. The investigations conclude that the resulting deformations of the cross-sections are very small and that the lift, drag and moment coefficients stay almost identical when compared to the respective values of the undeformed blade. These results indicate that cross-sectional deformations may be only of secondary importance for the aero-elastic behaviour of a wind turbine, and it seems therefore not essential to involve heavy FEM computations into the present aero-elastic modelling. Additionally, using tools like the beam cross section analysis software BECAS [26] makes it possible to appropriately describe the structural behaviour of a wind turbine blade with complex geometry and elaborate composite lay-up by only employing the computationally cheap beam-based structural models that are implemented in the popular aero-elastic simulation codes mentioned above.

Detailed stand-alone CFD rotor simulations have been employed in the field of wind energy since the late nineties [27] [28] also including the aerodynamic influences of tower and nacelle using overset meshes [29]. In conjunction with the wind tunnel test of the NREL phase VI rotor [30] a blind test could demonstrate for the first time that CFD gives viable results in wind turbine applications [31]. With the increase of computational power unsymmetrical load cases considering yaw conditions or wind shear [32] could be simulated using large CFD meshes of the entire rotor. A further increase in mesh size was necessary to conduct detached-eddy simulations (DES) for an improved wake modelling [33] and large eddy simulations (LES) for acoustical investigations [34]. More recently, Zahle [35] investigated the rotor-tower interaction of the NREL phase VI turbine using an overset grid method and Bechmann [36] conducted full rotor simulations of the MEXICO rotor [37] in order to compare the computations with the experiment and to extract data needed for simpler wake models. However, all the listed CFD computations assume a stiff structure and focus on the aerodynamic modelling only. Eventual effects on the structure or respective interactions with the structure are neglected.

The increase in computational power makes it meanwhile attractive to conduct

high-fidelity FSI simulations of an entire wind turbine using the coupling between both a high-fidelity FEM solver and a high-fidelity CFD solver. The increased interest in wind energy transmits this attraction to researchers which come from outside the traditional wind energy research but have the necessary stand-alone simulation tools available. As a consequence several publications can be found where mostly commercial software packages are used to establish a respective FSI coupling. However, the author found it hard to obtain substantial documentation and results which make it possible to better assess the value of the respective works. An exception is here the interesting work of Bazilev and Hsu [38] where the aero-elastic response of a full wind turbine with tower and nacelle is simulated using a finite element based ALE-VMS technique for the aerodynamics [39] and a NURBS based isogeometric analysis for the structure [40].

1.2.3 FSI at DTU Wind Energy

At the aero-elastic design section of DTU Wind Energy (Campus Risø) which currently has a leading position in the aeroelasticity research of wind turbines [19] various tools have been developed to describe the aero-elastic response of both two-dimensional aerofoil sections, isolated blades and entire wind turbines. The present section will give a closer look into the respective activities of the department and will particularly discuss the aerodynamic models employed in recent aero-elastic investigations. This will give a better understanding of the expected benefits and thus the motivation of establishing the high-fidelity FSI coupling between the aero-elastic solver HAWC2 [1] and the CFD solver EllipSys3D [2], [3], [4] which were both developed at the predecessor departments of DTU Wind Energy.

Low-Fidelity FSI Computations in 2D and 3D

The term *low-fidelity* is used to describe models which utilize a so-called engineering method to formulate the aerodynamics. In Scheepers [41] such a low-fidelity engineering model is defined as a model which *casts a complicated flow phenomenon into a transparent form.* Compared to high-fidelity CFD computations the low-fidelity formulations exhibit a decreased level of complexity and are thus computationally less expensive. Certainly, this decrease in complexity can also decrease the level of accuracy, however, very sophisticated low-fidelity models exist and their good performance can e.g. be examined in the work of Scheep-

ers, where the respective models are profoundly discussed. The term *low-fidelity model* should thus by no means be understood as a synonym for a retarded model.

Low-fidelity aerodynamic models are employed in all popular FSI codes for wind turbines. Like most other codes that predict the aero-elastic response of the entire wind turbine rotor, the FSI code HAWC2 developed at Risø DTU [1] employs the BEM theory to describe the three-dimensional aerodynamics around an entire wind turbine rotor. The theory is based on a one-dimensional momentum equilibrium which is established independently for each concentric annular element of the rotor. At each concentric element the flow is considered to be two-dimensional, meaning that two-dimensional models can be utilized to describe the respective aerodynamics. The following two-dimensional low-fidelity aerodynamic models were developed and then implemented in HAWC2.

- The aerodynamic model developed by Hansen [42] is a Beddoes-Leishman type of model and can predict the unsteady aerodynamic forces on a two-dimensional aerofoil section undergoing arbitrary motion in heave, lead-lag and pitch. In the attached flow regime the unsteady lift is computed using Theodorsen's theory [43]. In the separated flow regime the dynamic stall effects due to trailing edge separation are modelled by computing a weighted sum between the fully attached and the fully separated lift coefficients. When coupling this aerodynamic model to a structural model in a time marching aero-elastic simulation the governing differential equations can be solved very efficiently by using Duhamel's superposition integral [44].

- The aerodynamic model of Gaunaa [45] is based on potential flow assumptions and thin aerofoil theory and can predict the unsteady aerodynamics of a moving aerofoil section with variable geometry. As the model of Hansen [42] it employs Theodorsen's theory to calculate the unsteady lift forces in the attached flow regime and Duhamel's superposition integral is used to efficiently compute the unsteady circulatory lift components. The model can be employed to investigate aerofoils of variable geometry such as aerofoils equipped with trailing edge flaps or other active load reduction devices. Bergami [46] extended the validity of Gaunaa's model into the separated flow region by consolidating it with the dynamic stall model of Hansen. Since the above mentioned aerodynamic models are employing thin aerofoil theory it is shown in Bergami [47] how the indicial lift response function of the Duhamel integral can be conveniently derived for finite-thickness

aerofoils as well.

Since the BEM theory is based on a one-dimensional momentum equillibrium several corrections have to be implemented in order to account for the unsteady and three-dimensional effects which are experienced by the wind turbine rotor. Respective corrections account for tip losses, for dynamic inflow effects, for the unsymmetrical inflow during yaw and shear and for far and near wake effects [48], [49], [50], [51]. It also has to be considered that the aerodynamic forces determined by the models of Hansen, Gaunaa and Bergami are derived from tabulated two-dimensional aerofoil data. In a three-dimensional context this aerofoil data has to be corrected in order to account for the arising three-dimensional effects on the rotating wind turbine blade. Certainly, those corrections can only approximate the existing effects [52].

Apart from the implementation in the three-dimensional FSI computations of HAWC2 the aerodynamic model of Gaunaa [45] was also utilized in Buhl [53] in order to conduct two-dimensional FSI computations. In his study the aerodynamic model was coupled to a simple three degrees of freedom (3-DOF) structural model in order to investigate the aero-elastic response of an aerofoil section equipped with trailing edge flaps.

Both the two-dimensional FSI computation of an aerofoil section using the model of Buhl and the three-dimensional FSI computation of an entire wind turbine using the aero-elastic code HAWC2 are computationally very efficient. Due to the very efficient aerodynamic and structural model [1] of HAWC2 a respective FSI simulation can be carried out in real time on a standard desktop PC. This low demand on computational infrastructure is a key feature for engineering companies designing new wind turbines with HAWC2 and should not be underestimated when taking high-fidelity and computationally expensive alternatives into consideration.

Increased Reliability and Accuracy by using High-Fidelity CFD

The computational efficiency of the low-fidelity FSI simulation tools is only feasible by employing aerodynamic models that include the previously discussed simplifications and corrections. It is thus important to validate or verify the re-

[1] The structural model of HAWC2 is based on a multi-body formulation where the bodies are composed out of linear Timoshenko beam elements

spective models using measurement results or high-fidelity CFD computations. In both the work of Hansen [42] and the work of Bergami [46], [47] the aerodynamic models are compared with two-dimensional stand-alone CFD computations in order to demonstrate their very good performances. However, the models clearly have certain limits and only a certain range of validity in which reliable results can be expected. This can also be said for the BEM based aerodynamic model of HAWC2 where several operational conditions like the highly asymmetric loading in half wake situations [54] or the extreme angles of attack during stand-still or during an emergency shut-down are not expected to be modelled accurately.
In order to accurately model the aero-elastic load cases that exceed the limits of the low-fidelity aerodynamic models it is thus desirable to also have the opportunity of including high-fidelity CFD computations into the aero-elastic investigation process. CFD models like the finite volume Navier-Stokes solvers EllipSys2D and EllipSys3D do not exhibit the limitations of the previously described low-fidelity aerodynamic models and can thus contribute to an FSI simulation of greatest accuracy and detail. CFD computations can provide reliable results especially in flow regimes where viscous effects play an important role, and they can provide a lot of additional information such as e.g. the sectional pressure and skin friction distributions along the blade span. Three-dimensional full rotor CFD simulations are able to model the rotor wake and can e.g. provide additional information about the limiting streamlines on the rotor blades.

However, using CFD for the aerodynamic modelling increases the computational expenses of an FSI simulation decisively and the computation of a three minutes aero-elastic response of a wind turbine can no longer be accomplished in real time on a standard desktop PC but needs to be run for several hours or days on a huge computer cluster. Nevertheless, the aerodynamic details revealed in such a high-fidelity FSI computation can give valuable new insights to simulation test cases in which the traditional aerodynamic models might not be able to provide reliable results. Apart from that a high-fidelity FSI simulation tool using CFD will be very useful in order to verify the existing low-fidelity aero-elastic simulation tools which will certainly retain their importance within the aero-elastic design process.

Two-dimensional aero-elastic simulations using CFD have been recently carried out in Heinz [55] where the two-dimensional CFD solver EllipSys2D was coupled with the 3-DOF structural solver of Buhl [53] in order to investigate the aero-

servo-elastic response of a two-dimensional aerofoil section equipped with trailing edge flaps. Three-dimensional aero-elastic simulations using CFD have also been carried out at the aero-elastic design section of DTU Wind Energy. In Bertagnoli [56] the aerodynamic damping and the aero-elastic response of an isolated blade was investigated by using the structural response of HAWC [57][2] and the aerodynamic forces of EllipSys3D. However, since the work of Bertagnoli pursued a monolithic approach and employed the old and obsolete aero-elastic code HAWC an entirely new approach was preferred for the present work.

1.3 Scope of the Present Work

The present work has the aim of developing a high-fidelity FSI simulation tool that computes the aero-elastic response of an entire wind turbine rotor by coupling the structural model of HAWC2 with the aerodynamics of the three-dimensional flow solver EllipSys3D. Employing a high-fidelity flow solver for the three-dimensional modelling of the wind turbine rotor will eliminate the several limitations of the low-fidelity aerodynamic models of the traditional HAWC2 solver mentioned in Section 1.2.3 and will lead to a decisive improvement of the modelling accuracy. The computationally heavy investigations of the new model will then also help to verify the efficient low-fidelity models. The new FSI simulation tool will employ the efficient multi-body based formulation of the traditional HAWC2 solver since at the present state it was not considered to be meaningful to incorporate a high-fidelity FEM model of an entire wind turbine into the aero-elastic investigations. As mentioned in Section 1.2.2 it was demonstrated in the work of Kallesøe [25] that a detailed and computationally heavy FEM representation of the wind turbine structure is not expected to have a significant impact on the aerodynamics and thus on the aero-elastic response of the wind turbine model. The FSI coupling will be realized by following the partitioned coupling approach discussed in Section 1.2.1. The coupling will thus maintain the stand-alone solvers HAWC2 and EllipSys3D as independent entities both proven to provide reliable results within their respective fields of application. The coupling framework that organizes the data exchange between the solvers will be designed in a general way which makes it possible to also connect other stand-alone solvers.

The outline of the present work is as follows. In Chapter 2 the models that

[2] HAWC is the predecessor code of HAWC2

participate in the developed FSI simulation tool are presented. Apart from the aero-elastic solver HAWC2 and the high-fidelity CFD solver EllipSys3D several other models are presented. Those models have been connected to the same coupling framework in order to conduct aero-elastic and aero-servo-elastic computations of two-dimensional and three-dimensional aerofoil sections as well. Chapter 3 presents different coupling methods discussed in literature in order to establish a partitioned coupling of sufficient time accuracy and sufficient numerical stability. The chapter focuses on the recent findings in the field of aeroelasticity and relates them to the FSI problem of the present work. Chapter 4 presents the Python coupling framework which was developed to organize the execution and communication of the participating solvers. The framework is designed in a general way in order to easily establish various partitioned couplings of different solvers and different coupling methods. Chapter 5 focuses on the FSI coupling between HAWC2 and EllipSys3D. Specific issues considering the implemented force and deflection transfer as well as the implemented partitioned coupling schemes are discussed pursuing the aim of establishing a second order time accurate and sufficiently stable FSI simulation tool. Chapter 6 discusses specific issues of the FSI coupling that employs the simple three degrees of freedom (3-DOF) structural solver in order to investigate the aero-elastic behaviour of a two-dimensional or three-dimensional aerofoil section. Results of the respective FSI simulations have been submitted for publication in the journal *Wind Energy*. Chapter 7 presents the results of the first high-fidelity full rotor FSI simulations where the aero-elastic and even the aero-servo-elastic behaviour of the NREL 5MW reference turbine [58] is modelled. After assessing the performance of several implemented coupling schemes and after demonstrating that the favoured loose coupling scheme is sufficiently stable and second order accurate in time, the aero-servo-elastic response of the NREL 5MW reference wind turbine is modelled during different but constant inflow velocities in order to get a detailed picture of the respective baseline characteristics. Finally, the aerodynamically complex test case of an emergency shut-down is simulated and compared to the results of a traditional HAWC2 computation. The structural response of the simulated turbine includes tower, nacelle and drivetrain response and employs the variable speed, variable pitch controller documented in Jonkman [58]. Chapter 8 comprises the conclusion and an outlook.

Chapter 2

Participating Models

2.1 HAWC2

HAWC2 is an aero-elastic simulation tool for wind turbines developed at Risø DTU [1]. In its stand-alone version the aerodynamic model is based on the blade element momentum (BEM) theory and includes corrections to account for several unsteady and three-dimensional effects. The corrections incorporate dynamic stall, dynamic inflow, yaw and wake modelling [42], [48], [49], [50], [51]. Details about the underlying aerodynamic modelling have been given in Section 1.2.3. HAWC2 is widely used in industry and is a state-of-the-art aero-elastic code in order to develop and certify new wind turbine designs. Examples for its excellent performance can be found in Passon [59] and Larsen [60]. The code is under continuous development and is capable to compute the aero-elastic response of various wind turbine models. This not only includes the response of rather traditional two and three bladed onshore wind turbines but also comprises the aero-hydro-elastic response of rather new offshore concepts with mono-pile, tripod or jacket structures, and floating wind turbines fixed with mooring lines. The equations of motion are time integrated using the popular second order accurate and unconditionally stable Newmark algorithm as discussed in Krenk [61] with the Newmark parameters $\alpha = 1/2$ and $\beta = 1/4$.

It is the structural model of HAWC2 that will be employed in the high-fidelity FSI coupling between HAWC2 and EllipSys3D. This model is based on a multi-body formulation with bodies composed of one-dimensional beam structures. The relative positions and motions between the bodies are defined via certain constraints which can either define a fixed joint or a bearing that restrains the relative motion

in some degrees of freedom only. It is also possible to define a prescribed relative motion in order to e.g. describe the controlled pitch motion of the rotor blades. The multi-body formulation is able to appropriately model the large deflections occurring on modern wind turbines with long and flexible blades. This is different to other aero-elastic codes which exclusively use simple linear beam theory to model the structure. Linear beam theory is only valid for small deflection states and cannot account for the non-linear effects involved in the large deflections of modern wind turbine blades, and in contrast to the multi-body formulation the aerodynamic loads can here only be applied on the undeflected state.
In order to describe large blade deflections accurately HAWC2 is modelling the turbine blades by using several bodies that are connected with fixed joints. Assuming that an appropriate number of bodies is used to model the blades the relative deflections within the bodies are relatively small. This allows to model the internal deflections of the bodies with simple linear Timoshenko beam elements using e.g. the structural properties computed with BECAS [26] in order to accurately resemble the complex structural behaviour of the real composite blades. The respective approach is illustrated in Figure 2.1.
The figure further illustrates how the other structural components of a wind turbine are typically modelled in HAWC2, and since the FSI coupling between HAWC2 and EllipSys3D will employ the same type of structural model it was decided to discuss the respective build-up in more detail. The wind turbine tower is a rather stiff structure and the expected deflections are small. This is the reason why the tower is usually modelled by using only one body composed of several beam elements. The same counts for the shaft which is also modelled sufficiently by using only one body. The body is connected to the tower top by a bearing constraint that allows the shaft to rotate. The shaft is also split into several beam elements and the respective beam element nodes are positioned at the gearbox and at the main bearing in order to be able to extract and impose information about the actual positions and forces at the given points. The nacelle with its respective moments of inertia is modelled as a simple point mass located at the tower top. Since the hub of a real wind turbine has a certain diameter the blades are mounted with a certain distance to the rotational centre of the shaft. In order to implement this configuration in the structural model of HAWC2 a massless and infinitely stiff hub body is attached to the end node of the shaft. On top of this hub element the blade is attached using a bearing constraint that allows the introduction of a prescribed pitch motion.

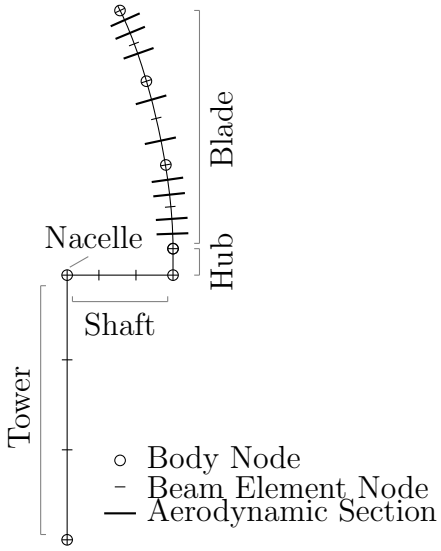


Figure 2.1: Typical Components of a Wind Turbine Model in HAWC2

In the stand-alone version of HAWC2 the aerodynamic loads are applied to the structure via radially distributed aerodynamic sections oriented perpendicular to the elastic axis of the blades. As illustrated in Figure 2.1 the radial positions of those aerodynamic sections are not related to the positions of the beam element nodes, instead, they are distributed in a way that assures a high density at areas with high force gradients. After the aerodynamic forces are determined at the respective aerodynamic sections using BEM theory they are linearly interpolated and then transferred to the beam element nodes where they are finally applied to the structure.

Customized functionality can be added to HAWC2 by using external models which, depending on the chosen operating system, are either provided with external dynamic link libraries or with external shared object files.[1] This allows the end-user to e.g. define customized external loadings on the turbine structure or to add customized control models to the aero-elastic simulation. Recently the interface could be used to implement external libraries for the modelling of mooring lines and soil spring forces for the simulation of offshore wind turbines.

2.1.1 Control Models

It is mentioned above that the functionality of HAWC2 can be extended with customized control models using dynamic link libraries or shared object files

[1] The HAWC2 code is originally developed for Windows, however, for the present work, the code was modified in order to also run on a Linux platform.

respectively. This makes it possible to e.g. control the generator speed and the pitch angles of the wind turbine and to thus conduct aero-servo-elastic simulations where the wind turbine model persistently runs in a realistic operational state. The control models can be used to e.g. describe and model several load cases of the IEC standards including the regular start-up and shut-down procedures but also to describe extreme events like an emergency shut-down after a loss of power. Since the models are not part of the actual solver and can be developed by the end-user the customer is able to implement and test its own customized control algorithms.

2.2 EllipSys3D

The high-fidelity CFD solver EllipSys3D was developed by Michelsen [2], [3] and Sørensen [4] and solves the incompressible Reynolds averaged Navier Stokes Equations (RANS) using primitive variables (u, v and p) in curvilinear coordinates through a multi-block finite volume discretization approach. The variables are stored in a collocated grid arrangement, and odd/even pressure decoupling is avoided using the Rhie-Chow interpolation [62]. For incompressible flow an additional equation is needed for the pressure and the standard practice is to derive a pressure equation (Poisson equation) by combining the continuity equation with the momentum equations. The momentum and pressure equations are then used in a predictor-corrector fashion to determine the pressure and velocities of the new time step where both the Semi-Implicit Method for Pressure-Linked Equations (SIMPLE) algorithm and the Pressure Implicit with Splitting of Operators (PISO) algorithm can be used. The convective terms are discretized using either the Second-Order Accurate Upwind Difference Scheme (SUDS) or the third order accurate Quadratic Upstream Interpolation for Convective Kinematics (QUICK) scheme. The viscous terms are discretized using the second order accurate central difference scheme (CDS). Further details about the above mentioned methods can be found in Ferziger [63]. For the present work the RANS equations were closed with the $k-\omega\,SST$ (Shear Stress Transport) turbulence model by Menter [64] which shows good performance for the adverse pressure gradients of aerofoil flows [65]. The time integration of the Navier Stokes Equations is accomplished by using the second order accurate three point backward difference scheme.

2.2.1 Second Order Accurate Moving Mesh Routine

A fundamental property for a CFD solver that participates in a FSI interaction is the capability of moving the grid points and boundary cells of the computational mesh in accordance to the deflections given by the structural solver.

In order to discuss how to integrate the moving mesh capabilities into the fluid solver the Navier-Stokes Equations are written in the so-called Eulerian formulation for fixed grids. Considering a certain control volume Ω_i and focusing only on the x-components the respective Navier-Stokes Equations can be written as

$$\rho \cdot \frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega_i} u_{x,i} \, \mathrm{d}\Omega_i \; + \; \rho \cdot \int_{\Omega_{ij}} u_{x,i} \mathbf{u}_i \boldsymbol{\mu}_{ij} \, \mathrm{d}\Omega_{ij} = g_{x,i}^{EUL}(\mathbf{u}_i, \mathbf{x}) \tag{2.1}$$

$$\rho \cdot \frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega_i} \mathrm{d}\Omega_i \; + \; \rho \cdot \int_{\Omega_{ij}} \mathbf{u}_i \boldsymbol{\mu}_{ij} \, \mathrm{d}\Omega_{ij} = 0 \tag{2.2}$$

where $\mathbf{u_i}$ is the velocity vector, $\mathbf{x}$ are the grid point positions of the mesh, Ω_{ij} is the cell face to the neighbouring cell j, $\boldsymbol{\mu}_{ij}$ is the normal vector of that cell face and $g_{x,i}^{EUL}$ represents the Eulerian diffusive fluxes and source terms.

The moving mesh capabilities can now be implemented into Equation 2.1 and Equation 2.2 by rewriting them in the so-called Arbitrary Lagrangian-Eulerian (ALE) form

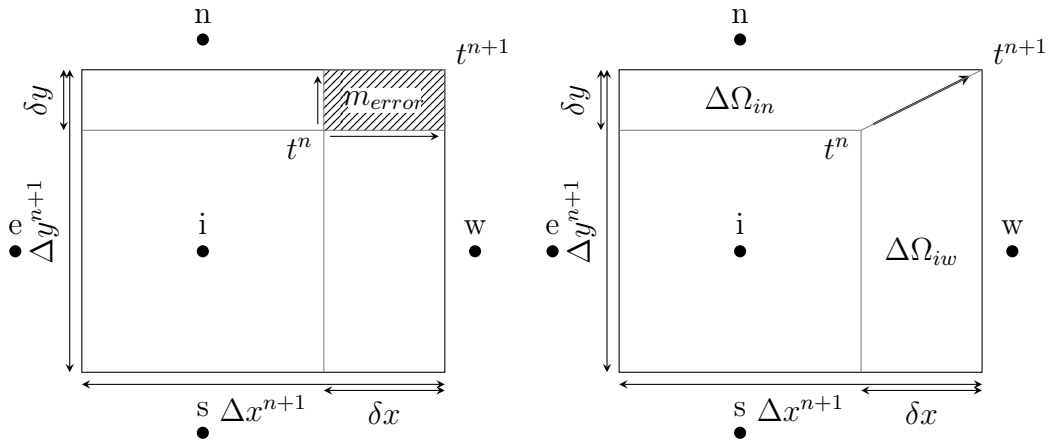
$$\rho \cdot \frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega_i} u_{x,i} \, \mathrm{d}\Omega_i \; + \; \rho \cdot \int_{\Omega_{ij}} u_{x,i} (\mathbf{u}_i - \mathbf{w}_i) \boldsymbol{\mu}_{ij} \, \mathrm{d}\Omega_{ij} = g_{x,i}^{ALE}(\mathbf{u}_i, \mathbf{x}) \tag{2.3}$$

$$\rho \cdot \frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega_i} \mathrm{d}\Omega_i \; + \; \rho \cdot \int_{\Omega_{ij}} (\mathbf{u}_i - \mathbf{w}_i) \boldsymbol{\mu}_{ij} \, \mathrm{d}\Omega_{ij} = 0 \tag{2.4}$$

where $g_{x,i}^{ALE}$ represents the ALE diffusive fluxes and source terms and where $\mathbf{w}_i$ is the velocity vector of the moving cell vertices. In case the mesh velocity is equal to zero ($\mathbf{w} = 0$) the formulas reduce to the Eulerian form of a fixed grid. In case the mesh velocity is equal to the fluid velocity ($\mathbf{w}_i = \mathbf{u}_i$) the control volume is moving with the fluid and always the same fluid particles remain inside. This corresponds to an entirely Lagrangian point of view.

In practice, however, it is rather difficult to determine the proper mesh velocity vector $\mathbf{w}$. In order to demonstrate this a simple test case is illustrated in Figure 2.2a where the mesh motion of a control volume with a unit depth of

$\Delta z^n = \Delta z^{n+1} = 1$ is shown during the time interval $[t^n,t^{n+1}]$. In this example the neighbouring cells j are indicated using compass notation i.e. $j = n, s, w, e$. For simplicity the mesh motion is limited to the east and north face only and the cell faces move with a constant speed w_x and w_y. The fluid velocities u_x and u_y are assumed to be the same in the entire control volume. Integrating the continuity equation 2.4 with the simple and first order accurate implicit Euler time integration scheme and applying it to the present test case gives



(a) Disrespecting GCL (b) Respecting GCL

Figure 2.2: Control Volume during Mesh Motion (With a Unit Depth of $\Delta z^n = \Delta z^{n+1} = 1$)

$$\begin{aligned}\frac{\rho(\Omega_i^{n+1} - \Omega_i^n)}{\Delta t} &+ \rho[(u_x - w_x)_w - (u_x - 0)_e]^{n+1}\Delta y^{n+1} \\ &+ \rho[(u_y - w_y)_n - (u_y - 0)_s]^{n+1}\Delta x^{n+1} = 0 \end{aligned} \tag{2.5}$$

The mesh velocities can be expressed with $w_x = \delta x/\Delta t$ and $w_y = \delta y/\Delta t$ and the mesh volumes of the two successive time instants can be calculated with $\Omega_i^{n+1} = \Delta x^{n+1}\Delta y^{n+1}$ and $\Omega_i^n = (\Delta x^{n+1} - \delta x)(\Delta y^{n+1} - \delta y)$. Plugging these expressions into Equation 2.5 shows that continuity is actually not fulfilled in the present test case and that the erroneous mass source m_{error}

$$\frac{\rho\delta x\delta y}{\Delta t} = m_{error} \neq 0 \tag{2.6}$$

is left. Figure 2.2a illustrates clearly where this erroneous mass source is coming from and why it is problematic to use the mesh velocity $\mathbf{w}$ as an input to the Navier-Stokes equations.

In order to circumvent a violation of the continuity equation in moving mesh algorithms it is a common practice to fulfil an additional relation, i.e. the geometrical conservation law (GCL)

$$\rho \cdot \frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega_i} \mathrm{d}\Omega_i \; - \; \rho \cdot \int_{\Omega_{ij}} \mathbf{w}_i \boldsymbol{\mu}_{ij} \, \mathrm{d}\Omega_{ij} = 0 \tag{2.7}$$

which can be also interpreted as the continuity equation for a non-moving fluid. The GCL can be easily fulfilled if the second term of the equation is not expressed directly via $\mathbf{w}_i$ but via an equivalent expression for the volume change due to $\mathbf{w}_i$. Denoting this volume change with $\sum_{j \in V(i)} \bar{\kappa}_{ij}$ and again using the simple and first order accurate implicit Euler scheme to integrate the GCL of Equation 2.7 leads to

$$\frac{\Omega_i^{n+1} - \Omega_i^n}{\Delta t} \; - \sum_{j \in V(i)} \bar{\kappa}_{ij} = 0 \tag{2.8}$$

where $V(i)$ denotes the set of vertices connected to vertex i and where $\bar{\kappa}_{ij}$ is the volume change related to the volume swept by the cell face Ω_{ij} during the time interval $[t^n,t^{n+1}]$. It is illustrated in Figure 2.2b that for the present test case this volume change can be calculated with

$$\sum_{j \in V(i)} \bar{\kappa}_{ij} = \sum_{j \in V(i)} \frac{\Delta \Omega_{ij}^{n+1}}{\Delta t} \tag{2.9}$$

and that in this way continuity is fulfilled and no erroneous mass is generated.

The above expression for $\sum_{j \in V(i)} \bar{\kappa}_{ij}$ is derived using the first order accurate implicit Euler time integration scheme. More recent publications such as Geuzaine [66], Farhat [67] and Ferziger [63] emphasize that alternative expressions have to be used if the underlying fluid solver uses a higher order accurate time integration scheme and if this order of time accuracy should be maintained by the implemented moving mesh algorithm. For the three point backward difference scheme of EllipSys3D the authors propose the expression

$$\sum_{j \in V(i)} \bar{\kappa}_{ij} = \sum_{j \in V(i)} \frac{\frac{3}{2}\Delta \Omega_{ij}^{n+1} - \frac{1}{2}\Delta \Omega_{ij}^{n}}{\Delta t} \tag{2.10}$$

where $\Delta \Omega_{ij}^{n+1}/\Delta t$ is the volume change between the time instants t^n and t^{n+1} and where $\Delta \Omega_{ij}^{n}/\Delta t$ is the volume change between the time instants t^{n-1} and t^n.

For the simulations of the present work the volume change $\sum_{j \in V(i)} \bar{\kappa}_{ij}$ is calculated as proposed in Equation 2.10 and it is thus expected that EllipSys3D maintains its second order time accuracy during moving mesh computations.

The remainder of this section is used to demonstrate how the expression of 2.10 is incorporated into the momentum equation of Equation 2.3. It should be mentioned here that the chosen terms and notations are related to the work of Geuzaine [66] and Farhat [67] in order to facilitate a direct and easy comparison between the analysis of their work and the respective implementation in EllipSys3D. A further discussion of their work, their findings and suggestions can be found in Section 3.2.

Letting $|\cdot|$ denote the measure of the geometric quantity $(\cdot)$ and letting $f_{x,i}^{ALE}$ denote the ALE convective fluxes Equation 2.3 can be rewritten as

$$\rho \cdot \frac{\mathrm{d}}{\mathrm{d}t}|\Omega_i u_{x,i}| \; + \; \rho \cdot f_{x,i}^{ALE}(\mathbf{u}_i, \mathbf{x}) = g_{x,i}^{ALE}(\mathbf{u}_i, \mathbf{x}) \tag{2.11}$$

Employing the three point backwards difference scheme to resolve the time integration of 2.11 results in

$$\begin{aligned} \rho \cdot (\frac{3}{2}(|\Omega_i|u_{x,i})^{n+1} - 2(|\Omega_i|u_{x,i})^{n} + \frac{1}{2}(|\Omega_i|u_{x,i})^{n-1}) \\ + \; \rho \cdot \Delta t \cdot f_{x,i}^{ALE}(u_{x,i}^{n^*}, \mathbf{u}_i^{n+1}, \bar{\mathbf{x}}_f) = \Delta t \cdot g_{x,i}^{ALE}(\mathbf{u}_i^{n+1}, \bar{\mathbf{x}}_g) \end{aligned} \tag{2.12}$$

where the ALE convective flux writes

$$\begin{aligned} f_{x,i}^{ALE}(u_{x,i}^{n^*}, \mathbf{u}_i^{n+1}, \bar{\mathbf{x}}_f) &= u_{x,i}^{n^*} \left(\sum_{j \in V(i)} \mathbf{u}_i^{n+1} \bar{\boldsymbol{\mu}}_{ij} \bar{\Omega}_{ij} - \sum_{j \in V(i)} \mathbf{w}_i \bar{\boldsymbol{\mu}}_{ij} \bar{\Omega}_{ij} \right) \\ &= u_{x,i}^{n^*} \left(\sum_{j \in V(i)} \mathbf{u}_i^{n+1} \bar{\boldsymbol{\nu}}_{ij} \quad - \sum_{j \in V(i)} \bar{\kappa}_{ij} \right) \end{aligned} \tag{2.13}$$

The bars above the geometrical quantities indicate that due to the mesh motion those values are time dependent and that a certain averaging has to be chosen. It is the central task of the work of Geuzaine [66] and Farhat [67] to determine those quantities in an appropriate manner such that the time accuracy of the fixed grid solution (see the Euler equations 2.1 and 2.2) can also be achieved when solving

the ALE equations 2.3 and 2.4.

In order to maintain the second order time accuracy of the CFD solver EllipSys3D the averaged quantities $\bar{\boldsymbol{\nu}}_{ij}$, $\bar{\kappa}_{ij}$ and $\bar{\mathbf{x}}_g$ should be chosen as given in Section 3.2.6.

Remark 1:
In EllipSys3D the velocity $u_{x,i}^{n^*}$ is only at the first subiteration chosen to be the value of the old time step t^n. After each subiteration i the value is then updated with the actual value of $u_{x,i}^{n+1}$. This treatment is chosen in EllipSys3D in order to linearize the otherwise non-linear expression of the convective fluxes.

Remark 2:
The quantities ν_{ij} and κ_{ij} of Equation 2.13 are defined as in the appendix of Geuzaine [66]. Slightly different and inconsistent definitions can be found in Farhat [67].

2.2.2 Parallelization

The CFD code EllipSys3D is using a multi-block decomposition (see Sørensen [4]) able to partition the computational domain into several sub-domains. The individual blocks can then be distributed to different computational nodes in order to accelerate the computation process. The communication between the blocks is done through a layer of ghost-cells located around each block and is technically accomplished by using the Message Passing Interface (MPI) library. In order to exploit the parallelization capability of EllipSys3D in a coupled FSI simulation as well, the Python coupling framework presented in Chapter 4 needs to support a parallel execution process as well.

2.3 Other Models

The coupling between HAWC2 and EllipSys3D provides the possibility of investigating the aero-elastic response of an entire wind turbine. By introducing some minor changes to the test set-up this coupling can be also used to simulate the aero-elastic response of an isolated blade only.

However, in the course of the present work the developed coupling framework of

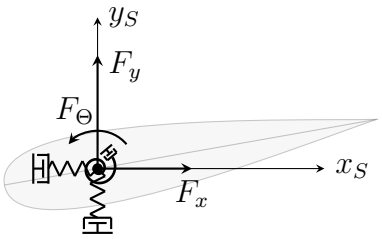


Figure 2.3: The 3-DOF Structural Solver (Connected to a $2D$ Aerofoil Section)

Chapter 4 was also used to realize aero-elastic simulations of both two-dimensional and three-dimensional aerofoil sections. The respective FSI coupling employs a simple three degrees of freedom (3-DOF) structural solver together with both the two-dimensional fluid solver EllipSys2D and the three-dimensional fluid solver EllipSys3D. In a first application Skrzypiński employed this coupling to investigate both stall-induced and vortex-induced vibrations of wind turbine blades during standstill [68], [69]. Together with the controllers presented in Section 2.3.3 the aero-elastic simulation tool could be extended to conduct aero-servo-elastic computations of a two-dimensional aerofoil section equipped with trailing edge flaps (such as presented in Heinz [55]).

2.3.1 3-DOF Structural Model

A simple 3-DOF structural model was implemented in order to facilitate aero-elastic computations of both two-dimensional and three-dimensional aerofoil sections. As illustrated in Figure 2.3 the model allows the aerofoil section to perform translational motions along the structural axis x_S and y_S and a rotation around the origin of the structural axis. Each degree of freedom is linked to a certain spring stiffness and a certain damping coefficient in order to mimic the structural properties at a given radial blade position. The model receives the three integrated force components lift, drag and pitch moment of the entire aerofoil section and applies them on the rotational centre of the structure. The resulting deflections are transferred back to the fluid solver. The respective equations of motion are time integrated using the explicit fourth order accurate Runge-Kutta-Nyström as suggested in Øye [70]. Further details about the implemented structural model together with reasonable choices for the structural quantities can be found in Heinz [55].

2.3.2 EllipSys2D

The CFD solver EllipSys2D is the two-dimensional complement to EllipSys3D. Two-dimensional simulations of aerofoil sections reduce the computational costs considerably when compared to three-dimensional computations and many aerodynamic effects can indeed be modelled adequately in two dimensions. However, for investigations like in Skrzypiński [68], [69] with deeply stalled and highly turbulent flow patterns an accurate simulation should comprise the involved three-dimensional effects as well.

2.3.3 Basic PI Flap Controllers

In Heinz [55] the FSI coupling between the 3-DOF structural model and EllipSys2D was used to estimate the load reduction potential of an aerofoil section equipped with trailing edge flaps. The work employed two simple PI controllers which have been implemented in order to control the flaps with respect to either a change in inflow angle or a change in the pressure difference between the suction and pressure side of the aerofoil. Both controllers have been connected to the Python coupling framework of Chapter 4 in order to conduct aero-servo-elastic investigations.

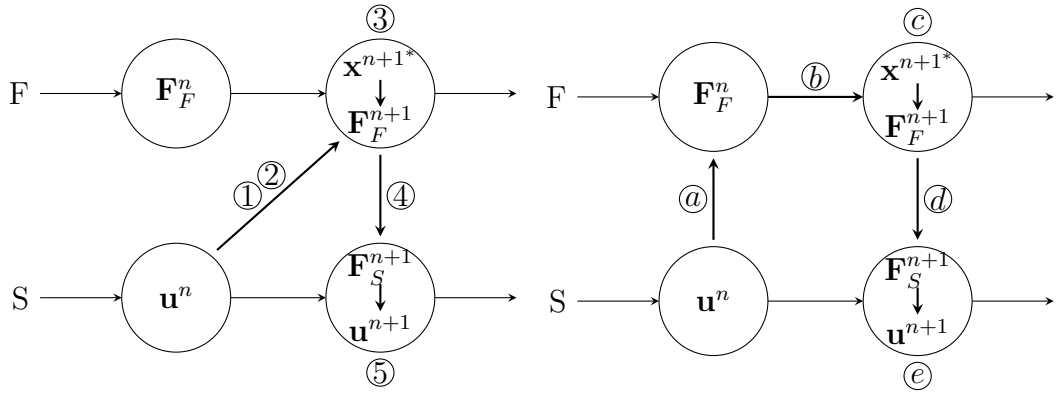
Chapter 3

Partitioned FSI in Good Quality

3.1 Strong vs. Loose Coupling

In Section 1.2.1 it was mentioned that partitioned coupling schemes have numerous advantages in comparison with monolithic approaches. However, partitioned coupling methods need to be designed wisely and with good care since they bear the risk of decreasing the time accuracy and the numerical stability of the coupled simulation.

A relatively simple and thus the most common coupling method for partitioned systems is the loose coupling method in which the participating models exchange their solutions only once per time step. Apart from a minimized data transfer, a loose coupling also minimizes the computational costs since each connected subsystem needs to be solved only once per time step. Several variations of loosely coupled schemes exist and an obvious and very common choice for a loosely coupled algorithm is shown in Figure 3.1 where the side of the fluid solver is indicated with the letter F and the side of the structural solver is indicated with the letter S. Representation 1 and Representation 2 can both be found in literature and practically describe the same procedure. In Farhat [71] and Farhat [8] this algorithm is denoted as *conventional serial staggered* (CSS) algorithm.
As done consistently in the entire work the aerodynamic forces on the fluid side are denoted with $\mathbf{F}_F$ and the respective aerodynamic forces on the structure side are denoted with $\mathbf{F}_S$. The structural deflections of the structural mesh are gathered in the vector $\mathbf{u}$ and the respective structural deflections of the fluid mesh are gathered in the vector $\mathbf{x}$. Quantities of the actual time step are labelled with the superscript n, quantities of the subsequent time step are labelled with the su-



(a) Representation 1 (b) Representation 2

Figure 3.1: Conventional Serial Staggered (CSS) Scheme

perscript $n+1$. The asterisk indicates quantities that are based on the predicted deflection state.

The CSS algorithm shown in Figure 3.1a (Figure 3.1b) is characterized by the following step cycle:

1. (i.e. Step b) Predict the structural deflections of the new time step using information about the previous structural deflections.

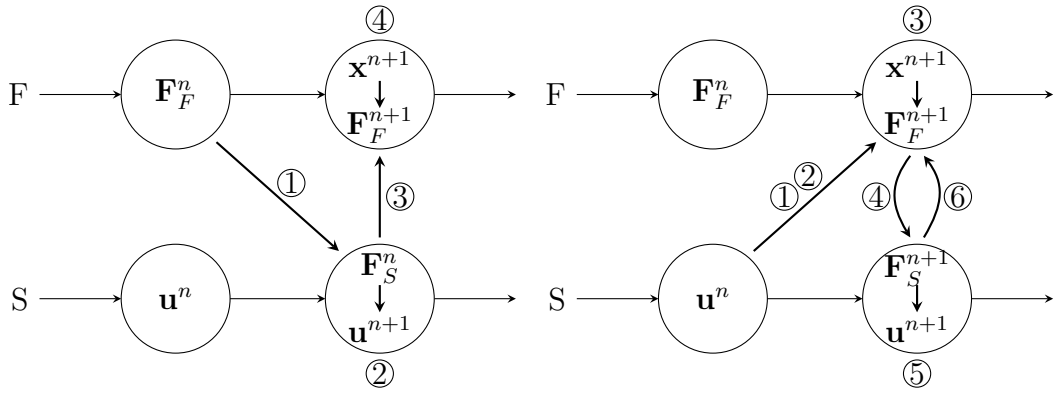
2. (i.e. Step a) Transfer the structural deflections from the structural solver to the fluid solver.

3. (i.e. Step c) Solve the fluid subdomain and calculate the new aerodynamic forces corresponding to the predicted deflection state.

4. (i.e. Step d) Transfer the new aerodynamic forces to the structural solver and apply them on the structures.

5. (i.e. Step e) Use the new aerodynamic forces and an implicit time integration to calculate the structural deflections of the new time step.

Unfortunately it was frequently reported that a basic implementation of the CSS scheme reduces the time accuracy of the coupled simulation at least with the order of one when compared to the original time accuracy of the connected stand-alone solvers [7]. At the same time it was reported that the stability limits of the coupled simulations can be reduced significantly as well. This potentially bad performance of the basic CSS schemes resulted in the bad reputation of loosely



(a) Elementary Loose Coupling Scheme (b) Strong Coupling Scheme

Figure 3.2: Elementary Loose Coupling and Strong Coupling Scheme

coupled schemes in general. However, recent developments in the field of aeroelasticity have identified reasons for the bad performance of the basic CSS algorithm and proposed several improved schemes in order to increase the numerical stability and the overall time accuracy of the coupled system [71], [7], [66], [67], [8]. In Section 3.2 those findings are discussed in more detail.

Another commonly used loose coupling scheme is shown in Figure 3.2a and is used in case the structural solver is equipped with an explicit time integration scheme only. This elementary coupling method also seems to be employed in several cases where less thoughts are given to the actual coupling method and where the underlying time integration schemes of the participating solvers are not understood.

The elementary coupling method of Figure 3.2a consists of the following step cycle:

1. Transfer the actual aerodynamic forces to the structural solver and apply them on the structures.

2. Use the actual aerodynamic forces and an explicit time integration in order to calculate the structural deflections of the new time step.

3. Transfer the new deflections from the structural solver to the fluid solver.

4. Solve the fluid subdomain and calculate the new aerodynamic forces corresponding to the new deflection state.

When compared to the basic implementation of the CSS scheme it can be seen that the elementary coupling scheme uses the actual (and not the predicted) deflection state in order to calculate the solution of the fluid domain. However, due to its explicit time integration in the structural subsystem the scheme is expected to have a worse performance in terms of time accuracy and stability.

Strong coupling schemes are generally a safer choice with respect to accuracy and stability and they are frequently used in case a simply designed loosely coupled scheme does not provide the desired results. In a strong coupling scheme the participating solvers exchange the structural deflections and the aerodynamic forces not only once per time step but also within their subiterations. This removes the explicit character of the CSS scheme where the deflections of the new time step are only predicted and exclusively based on the deflections of the previous time steps.
A typical strong coupling procedure is shown in Figure 3.2b and its respective step cycle is listed below. The scheme is derived from the CSS algorithm of Figure 3.1a and only differs in the iterative subcycling which commences with the additional step 6 and which is used to obtain a converged result between the actual deflections of the new time step and its corresponding aerodynamic forces. This is in contrast to the CSS algorithm where the aerodynamic forces are only based on the predicted deflections of step 1.

1. Predict the structural deflections of the new time step using information about the previous structural deflections.

2. Transfer the predicted deflections from the structural solver to the fluid solver.

3. Solve the fluid subdomain and calculate the new aerodynamic forces corresponding to the given deflection state.

4. Transfer the new aerodynamic forces to the structural solver and apply them on the structure.

5. Use the new aerodynamic forces and an implicit time integration to calculate the structural deflections of the new time step.

6. Transfer the new deflections back to the fluid solver and continue with step 3.

7. Subcycle step 3 to step 6 until convergence between forces and deflections is reached.

However, apart from a potentially more stable and more accurate simulation the computational costs of a strong coupling scheme increase significantly when both solvers have to compute multiple solutions per time step in order to reach convergence.

In order to optimize the computational costs and, at the same time, achieve a sufficient accuracy and stability of the coupled system it is thus advisable to first identify and understand the actual reasons that influence these attributes. The following two sections therefore discuss recent findings obtained in the fields of aeroelasticity and biomechanics; the two fields in which most research was done lately in terms of partitioned FSI computations. The presented insights will finally help to choose and adjust a suitable coupling algorithm for the desired FSI coupling between HAWC2 and EllipSys3D.

3.2 Loose Coupling in Aeroelasticity

In the field of aeroelasticity several studies can be found which successfully employ loosely coupled schemes for partitioned FSI simulations. In particular the work done by Farhat [71], [7], [66], [67], [8] identifies and eliminates several problems and shortcomings of traditional loosely coupled schemes. The authors blame those traditional and little understood schemes to be the reason for the bad reputation of loosely coupled systems in general and doubt that the alleged deficiencies of loosely coupled schemes always have to be fixed by computationally expensive subcycling. Instead they conclude that a wisely designed loosely coupled scheme can indeed provide computations of sufficient stability and sufficient time accuracy [8]. The several findings and improvements in order to achieve this goal are summarized and discussed within the following seven key components necessary to transform the basic CSS scheme of Figure 3.1 into a state-of-the-art *generalized serial staggered* (GSS) scheme, a term which is also introduced in the work of Farhat. The seven components are described from Section 3.2.1 to Section 3.2.7.

3.2.1 Force Transfer to Minimize Inherent Energy Inaccuracy

In Section 3.1 it was shortly mentioned that loosely coupled schemes inherently suffer from a lack of energy accuracy and are thus inherently prone to trigger numerical instabilities. This is due to the fact that the fluid solver is computing its solution based on a predicted deflection state while the structural solver is converging to the actual and slightly different deflection state. It is found in Piperno [7] that an adequate force transfer plays a major role in minimizing these inherent instabilities.

In order to obtain an energy conservative force transfer at the fluid-structure (FS) interface the energy released by the fluid has to be the same as the energy received by the structure. Let $\Delta E_{F,FS}$ and $\Delta E_{S,FS}$ be the respective energies released and received during the time interval $[t^n,t^{n+1}]$, let $\mathbf{x}_\Gamma$ be the mesh vertices of the deformed fluid solver on the common boundary surface Γ and let $\mathbf{u}_\Gamma$ be the respective deflection vector of the structural solver, we can express this claim for energy conservation with the following equation

$$\begin{aligned} \Delta E_{F,FS} &= \Delta E_{S,FS} \\ \bar{\mathbf{F}}_F \cdot (\mathbf{x}_\Gamma^{n+1} - \mathbf{x}_\Gamma^n) &= \bar{\mathbf{F}}_S \cdot (\mathbf{u}_\Gamma^{n+1} - \mathbf{u}_\Gamma^n) \end{aligned} \tag{3.1}$$

where $\bar{\mathbf{F}}_F$ are the aerodynamic forces acting in the fluid mesh and $\bar{\mathbf{F}}_S$ are the aerodynamic forces transferred to the structural mesh and acting on the structure.

The vectors $\bar{\mathbf{F}}_F$ and $\bar{\mathbf{F}}_S$ constitute a certain mean value of the forces within the time interval $[t^n,t^{n+1}]$ and depend directly on the employed time integration schemes of the participating fluid and structural solver. Assuming that both solvers employ a second order accurate time integration scheme the trapezoidal rule can be used to approximate the aerodynamic forces $\bar{\mathbf{F}}_F$ and $\bar{\mathbf{F}}_S$ during the time interval $[t^n,t^{n+1}]$ with

$$\bar{\mathbf{F}}_F = \frac{\mathbf{F}_F^{n+1} + \mathbf{F}_F^n}{2} \quad \text{and} \quad \bar{\mathbf{F}}_S = \frac{\mathbf{F}_S^{n+1} + \mathbf{F}_S^n}{2} \tag{3.2}$$

Considering a loose coupling, however, the mesh vertices $\mathbf{x}^{n+1}$ of the fluid solver are based on the predicted structural deflection $\mathbf{u}^{n+1^*}$ and not on the actual

converged deflections $\mathbf{u}^{n+1}$ of the structural solver, and since the solvers see a different structural motion the energies of Equation 3.1 will always be different and incomparable. A revised expression for the energy conservation of a loose coupling scheme that employs a second order accurate time integration scheme for both the fluid and the structural solver should thus state

$$\begin{gathered} \Delta {E_{F,FS}}^* \neq \Delta E_{S,FS} \\ (\frac{\mathbf{F}_F^{n+1} + \mathbf{F}_F^n}{2}) \cdot (\mathbf{x}_\Gamma^{n+1^*} - \mathbf{x}_\Gamma^{n^*}) \neq (\frac{\mathbf{F}_S^{n+1} + \mathbf{F}_S^n}{2}) \cdot (\mathbf{u}_\Gamma^{n+1} - \mathbf{u}_\Gamma^n) \end{gathered} \tag{3.3}$$

where the asterisk refers to the predicted deflection state and where the error in energy due to the different deflection states of a loose coupling can be written as

$$\Delta E_{Loose} = {\Delta E_{F,FS}}^* - \Delta E_{S,FS} \tag{3.4}$$

At the moment of the force transfer nearly all variables of Equation 3.3 and 3.4 are predetermined and have assigned values. The only variable that is still free to be determined is the force vector $\mathbf{F}_S^{n+1}$ used to subsequently compute the respective structural deflection state $\mathbf{u}_\Gamma^{n+1}$. Although it is not desired and even senseless to enforce the energies $\Delta E_{F,FS}^*$ and $\Delta E_{S,FS}$ to be equal, Piperno [7] could demonstrate that an appropriate choice of $\mathbf{F}_S^{n+1}$ could limit the introduced erroneous energy to a higher order effect. Therefore Piperno incorporates the expression of Equation 3.4 into an analytical investigation of an improved CSS algorithm and presents several optimal choices for $\mathbf{F}_S^{n+1}$ which depend on both the chosen time integration schemes of the fluid and structural solver and on the chosen structural predictor step.

Considering the second order accurate time integration schemes of HAWC2 and EllipSys3D, employing a second order time accurate structural prediction and assuming a energy conservative force and deflection transfer as discussed in Section 3.2.7 the simple relation of

$$\mathbf{F}_S^{n+1} = \mathbf{F}_F^{n+1} \tag{3.5}$$

is then proven to result in a third order energy accurate loosely coupled algorithm. Here, a third order energy accurate coupling scheme means that the error in energy ΔE_{Loose} is reduced by the factor of eight if the time step is halved.

3.2.2 Prediction of Structural Deflections

The GSS coupling scheme predicts the structural deflections of the new time step with

$$\mathbf{u}^{n+1^*} = \mathbf{u}^n + \alpha_0 \Delta t \dot{\mathbf{u}}^n + \alpha_1 \Delta t^2 \ddot{\mathbf{u}}^n \tag{3.6}$$

where $\mathbf{u}$ is the deflection vector and Δt is the time step size, and where the superscript n indicates the actual time step and the asterisk indicates the predicted state. In this generalized formulation the constant factors α_0 and α_1 are used in order to adjust the formula to an either second order accurate ($\alpha_0 = 1$, $\alpha_1 = 1/2$), first order accurate ($\alpha_0 = 1$, $\alpha_1 = 0$) or trivial ($\alpha_0 = 0$, $\alpha_1 = 0$) prediction.
In the work of Piperno et. al [7] it was shown that the chosen accuracy of the prediction step is decisive for minimizing the erroneous energy ΔE_{Loose} of Equation 3.4 and that a consistent choice of α_0 and α_1 could reduce the error to a third order effect only.
In the work of Farhat et al [8] the importance of a suitable choice for the factors α_0 and α_1 was again put into focus when a second order prediction of the deflection was proven to be crucial in order to establish a second-order time accurate solution algorithm.

Considering the second order accurate time integration schemes of HAWC2 and EllipSys3D a second order prediction with $\alpha_0 = 1$ and $\alpha_1 = 1/2$ would be needed in order to obtain a coupling which maintains the second order accuracy in time and which limits the erroneous energy ΔE_{Loose} to a third order effect only.

3.2.3 Mesh Deformation Matrix

It is a common procedure to formulate an FSI problem in a so-called three field problem. This means that apart from the two fields for fluid and structure a third set of equations is formulated which is used to describe the dynamic mesh motion inside the fluid solver. The mesh motion is then modelled as a pseudo-structure subsystem where a usually time dependent fictitious stiffness matrix $\mathbf{U}(t)$ is used to provide a smooth deformation of the CFD mesh.

In order to deform the CFD mesh in correspondence to the predicted deformation state $\mathbf{u}^{n+1^*}$ of the structural solver the authors of [8] formulated the following equation

$$\mathbf{x^{n+1^*}} = \mathbf{x^{n^*}} + \bar{\mathbf{U}}(\mathbf{u_\Gamma^{n+1^*}} - \mathbf{u_\Gamma^{n^*}}) \tag{3.7}$$

where the vector $\mathbf{x}^{n+1^*}$ contains the grid points of the CFD mesh describing the predicted deflection state of the new time step and the matrix $\bar{\mathbf{U}}$ is a mean value of the mesh deformation matrix $\mathbf{U}(t)$ within the time interval $[t^n, t^{n+1}]$.

It is shown in [8] that a suitable choice of the matrix $\bar{\mathbf{U}}$ is essential to establish a higher order accurate coupling scheme, and it is emphasized that to the best of the authors knowledge nobody before was formulating the problem in this way and thus nobody before was aware of it. In contrast to the usual practice of calculating the new mesh points by just using the mesh deformation matrix of the new time step (i.e. $\bar{\mathbf{U}} = \mathbf{U}^{n+1}$), it is proven in [8] that an appropriately chosen mesh deformation matrix of $\bar{\mathbf{U}} = (\mathbf{U}^{n+1} - \mathbf{U}^n)/2$ is needed to establish the desired second order accurate coupling scheme.

Remark:
In Equation 3.7 the notation $\mathbf{u}_\Gamma$ is used to indicate the grid points of the structural mesh which are located on the wet surface Γ, i.e. the surface directly in contact with the computational grid of the flow solver. However, in the present coupling between HAWC2 and EllipSys3D the structural model employs one-dimensional beam elements only and does not directly provide the grid points at the wet surface. In Section 5.3.3 it is thus discussed in more detail how this problem is handled in the FSI coupling of the present work and why the respective mesh deformation matrix $\bar{\mathbf{U}}$ is not reducing the second order time accuracy of the overall coupling algorithm.

3.2.4 Time Integration Scheme of the Structural Solver

Certainly, the time integration scheme of the participating solvers play a decisive role in order to develop a suitable design for the GSS coupling scheme. First of all the chosen time integration scheme has to provide at least the same order of accuracy as the one expected from the coupled simulation. Secondly, the design

of several other GSS components depend on and vary with the time integration scheme used in the structural solver.

The structural solver HAWC2 employs the popular second order accurate and unconditionally stable Newmark algorithm as discussed in Krenk [61] where the Newmark parameters are set to $\alpha = 1/2$ and $\beta = 1/4$. Since the authors of [7] and [8] consider the very same time integration scheme the respective findings can be directly related to the present FSI coupling between HAWC2 and EllipSys3D.

3.2.5 Time Integration Scheme of the Fluid Solver

The impact of the time integration scheme of the fluid solver is comparable to the one of the structural solver. The time integration scheme of the fluid solver EllipSys3D is the popular and second order accurate three point backward difference scheme (see Equation 2.12). Since the authors of [7], [8], [66] and [67] utilize the same time integration scheme for the fluid solver (among several other schemes) it is possible to directly relate their findings to the FSI coupling between HAWC2 and EllipSys3D.

3.2.6 Calculation of ALE Fluxes

In order to establish an FSI coupling the participating CFD solver needs to be equipped with a moving mesh algorithm that allows the computational grid to be moved during the simulation. It was pointed out in the work of Geuzaine [66] and Farhat [67] that a careless implementation of those moving mesh capabilities can easily result in a degradation of the overall time accuracy.

As already discussed in Section 2.2.1 the implementation of a moving mesh algorithm leads to the problem of solving the Navier-Stokes equations in ALE formulation (see Equation 2.3 and 2.4). Geuzaine and Farhat propose different methodologies for solving the Navier-Stokes Equations in ALE formulation. They analytically investigate several possibilities to time average the time dependent mesh configurations and to thus average the related ALE convective and diffusive fluxes.

The most essential statement with respect to the fluid solver EllipSys3D is made as a remark at the end of the appendix in Geuzaine [66]. Here it is stated that

the so-called *ALE Scheme C* maintains the second order accuracy of the fluid solver if the ALE equations are solved by using the quantities

$$\begin{aligned} \bar{\boldsymbol{\nu}}_{ij} &= \boldsymbol{\nu}_{ij}^{n+1} \\ \bar{\kappa}_{ij} &= \frac{\frac{3}{2}\Delta\Omega_{ij}^{n+1} - \frac{1}{2}\Delta\Omega_{ij}^{n}}{\Delta t} \\ \bar{\mathbf{x}}_g &= \mathbf{x}_g^{n+1} \end{aligned} \tag{3.8}$$

The three quantities $\bar{\boldsymbol{\nu}}_{ij}$, $\bar{\kappa}_{ij}$ and $\bar{\mathbf{x}}_g$ have been introduced and explained in Section 2.2.1 and the respective values given in Equation 3.8 are exactly the choices made in the present implementation of the fluid solver EllipSys3D.

3.2.7 Conservative Force and Deflection Transfer at Interface Boundary

One major issue in an FSI coupling of good quality is the appropriate transfer of the forces and deflections. In Section 3.2.1 it was discussed how an adequate force transfer can minimize the inherent energy inaccuracy of a loose coupling scheme, a problem that was related to the temporal progression of a loosely coupled scheme and the chosen time discretizations in the connected solvers. The present chapter focuses on an adequate and thus energy conservative force and deflection transfer in terms of the different spatial discretizations usually encountered at the interface between the structure and the fluid mesh.

At the interface boundary Γ between the fluid and the structural domain the following two conditions are commonly used in order to ensure dynamic and kinematic continuity

$$\sigma_S \cdot \mathbf{n} = p \cdot \mathbf{n} + \sigma_F \cdot \mathbf{n} \quad \text{on} \quad \Gamma \tag{3.9}$$

$$\mathbf{u}_\Gamma = \mathbf{x}_\Gamma \quad \text{on} \quad \Gamma \tag{3.10}$$

with σ_S and σ_F being the structure stress tensor and the fluid viscous stress tensor, with $\mathbf{n}$ being the normal vector to Γ and with $\mathbf{u}_\Gamma$ and $\mathbf{x}_\Gamma$ being the displacement fields of structure and fluid on Γ. The dynamic boundary condition of Equation 3.9 states that the tractions on the wet surface of the structure are in

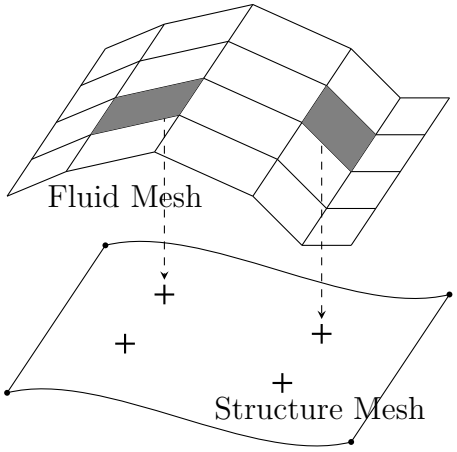


Figure 3.3: Non-Conservative Load Transfer at FS Interface

equilibrium with the tractions on the fluid side and thus leads to a conservation of momentum. The kinematic boundary condition of Equation 3.10 states that on the wet surface the displacement field of the structure corresponds to the displacement field of the fluid and thus leads to a conservation of mass. Fulfilling both conditions simultaneously finally leads to a conservation of energy.

Hence, the general opinion is that energy should be conserved when the forces and deflections are transferred via the fluid-structure (FS) interface. Usually the meshes of fluid and structural solver are non-matching at the FS interface and the chosen spatial discretization methods of the connected solvers are different as well. This means that special care has to be taken in order to establish a force and deflection transfer that fulfils the boundary conditions formulated above and thus conserves energy at the FS interface. Assuming a coupling between a CFD and a FEM solver a simple and obvious force transfer approach can be illustrated as done in Figure 3.3. This approach extracts the pressure values from those fluid cells that are located close-by the Gauss points of the structural mesh where the respective values are requested from the structure. However, since the spatial discretization and interpolation methods of the fluid and the structural mesh are different it can be clearly seen that the virtual work done by the fluid

$$\delta W_F = \mathbf{F}_F \cdot \delta \mathbf{x}_\Gamma \tag{3.11}$$

is not the same as the respective virtual work experienced by the structure

$$\delta W_S = \mathbf{F}_S \cdot \delta \mathbf{u}_\Gamma \tag{3.12}$$

and that such an approach is thus not energy conservative.

In Farhat [6] it is shown that energy conservation can be achieved if the displacements of the fluid mesh are expressed by using the discretization and thus the continuous shape functions of the structural solver. By expressing the displacements of the fluid mesh via

$$\delta \mathbf{x}_\Gamma = \boldsymbol{\eta}(\mathbf{x}_u) \cdot \delta \mathbf{u}_\Gamma \tag{3.13}$$

where $\boldsymbol{\eta}$ contains a certain set of shape or interpolation functions and where $\mathbf{x}_u$ contains the coordinates of the fluid points projected on the initial or undeformed structural mesh, and by equating the virtual works δW_F and δW_S

$$\mathbf{F}_F \cdot \delta \boldsymbol{\eta}(\mathbf{x}_u) \cdot \delta \mathbf{u}_\Gamma = \mathbf{F}_S \cdot \delta \mathbf{u}_\Gamma \tag{3.14}$$

it can be seen that the applied force vector $\mathbf{F}_S$ can be calculated with the transpose of the interpolation function matrix

$$\mathbf{F}_S = \boldsymbol{\eta}(\mathbf{x}_u)^T \cdot \mathbf{F}_F \tag{3.15}$$

in order to obtain an energy conservative load and motion transfer.

The energy conservative load and motion transfer as presented in Farhat still assumes the non-matching meshes to share at least a common wet surface (see Figure 5.9a). However, a similar approach can be found in the work of Brown [72] where the method is applied on highly non-matching meshes using three-dimensional aerofoil meshes in the CFD solver and certain wing box structures of one-dimensional beam elements in the structural solver. Instead of an interpolation function matrix $\boldsymbol{\eta}(\mathbf{x}_u)$ Brown utilized an extrapolation function $\mathbf{N}(\mathbf{x_u})$ to extrapolate the displacements of the beam elements to the displacement field of the three-dimensional CFD mesh.

An approach as discussed in Brown might be also applicable for the FSI coupling between HAWC2 and EllipSys3D where the structural model is built up using one-dimensional beam elements and where the CFD mesh models the three-dimensional rotor shape exactly (see Figure 5.9b). A respective draft can be found in Appendix B.4. However, the present implementation of the force and deflection transfer does not directly impose energy conservation. The chosen

non-conservative approach will be explained in Section 5.2 and will be further discussed in Section 5.3.1.

3.3 Strong Coupling in Biomechanics

While in the field of aeroelasticity a wisely designed loosely coupled scheme seems to have the potential to provide sufficiently stable and sufficiently time accurate results the preferences in the field of biomechanics clearly tend to use strongly coupled schemes.

In the publications of Mok [9], Wall [10], Causin [11] and Förster [12] the scepticism towards loosely coupled schemes is based on the so-called *artificial added mass effect* which can be observed during the simulation of incompressible fluids in case the continuity equations at the FS interface are slightly violated. A good description of the artificial added mass effect is given in [9]. The description commences with the explanation that the fluid pressure heavily depends on the inertia of the fluid. At the FS interface the fluid has to follow the motion of the structure and an inaccurate description of the mesh motion will quickly introduce erroneous accelerations, erroneous inertia and thus erroneous pressure forces. In an incompressible fluid this pressure is not dampened and will directly act on the structure where it will cause a distinct structural reaction and in return an even further increased inertia force. This amplification leads to numerical instabilities which are known as the artificial added mass effect. A particular problem is that this effect is further amplified if smaller time steps are used.

Loosely coupled schemes are particularly prone to trigger the artificial added mass effect since they inherently violate the continuity equations and inherently introduce inaccuracies and slightly incorrect coupling forces at the FS interface (see Section 3.2.1). In [9] the artificial added mass effect is suspected to be the probably only reason why loosely coupled schemes suffer from numerical instabilities. In Förster [12] it is argued that - without exception - all loosely coupled schemes suffer from numerical instabilities and that modification as suggested in Piperno [7] can only postpone the onset of instabilities but cannot prevent them. Several empirical studies and the mathematical observations in Causin [11] and Förster [12] lead to the following conclusions:

- The mass ratio between fluid and structure has a significant influence on

the stability of the sequentially staggered (i.e. loosely coupled) system. The bigger the mass ratio ρ_F/ρ_S the worse the instabilities get.

- The instabilities are less severe in case the fluid viscosity or the structural stiffness is increased.
- The chosen time integration scheme of the fluid solver is also found to be relevant. Unfortunately, a higher order time integration scheme such as the three point backwards scheme is more likely to trigger instabilities than the only first order accurate implicit Euler scheme.
- Decreasing time steps increase the instabilities.

Clearly, the above listed characteristics of loosely coupled algorithms are all very critical in terms of biomechanical FSI simulations and it is concluded in Wall [10] that the only appropriate choice of achieving a sufficiently stable partitioned algorithm is to introduce subiterations, i.e. employing a strongly coupled scheme. However, with respect to aero-elastic simulations it can be argued that at least the first two points of the above listed observations are much better fitting to aero-elastic problems where usually relatively heavy structures are investigated. In terms of instabilities it is obvious that the FSI simulation of a bloodstream through a thin and elastic vascular wall is much more delicate than the simulation of a wind turbine structure interacting with air.

Evaluating the different findings in literature and relating them to the intended coupling between HAWC2 and EllipSys3D in order to simulate the aero-elastic response of wind turbines it seems possible to establish a sufficiently stable and sufficiently time accurate FSI coupling by using a carefully designed loosely coupled scheme as discussed in Section 3.2. However, certain doubts and findings about several weaknesses of loosely coupled schemes led to the decision of also implementing a strongly coupled scheme into the present coupling framework. A final overview of the implemented coupling schemes will be given in Section 5.4. An assessment of the different schemes can then be found in Section 7.2.

Chapter 4

Python Coupling Framework

4.1 General Demands for the Coupling

In order to establish a partitioned coupling between the solvers presented in Chapter 2 a coupling framework is developed that controls the execution of the participating models and organizes the data transfer in-between them. Most of the considered solvers are stand-alone solvers which are originally developed in order to be executed individually. The coupling framework should thus respect and maintain the given code structures and be minimal intrusive in order to allow a further independent development of the respective codes.
Furthermore the framework should be formulated in a generic way which makes it possible to easily connect new models and to easily exchange existing models in the future as well. In order to maximize the number of potentially connectable solvers it is also desired to create a coupling framework that can incorporate solvers of many different programming languages and that can be executed on different operating systems.

4.2 Python Potentials

It was decided to develop the desired coupling framework in Python. Python is a high-level, object-oriented computer language known for its clear syntax and easy readability. It features compound and dynamic data types and is compiled during runtime, leading to a quick and convenient way of programming. These attributes fit well to the above mentioned demands for an easy and quick manipulation of the coupling. Since Python is under an open source license and free to use a vast amount of additional packages and libraries are developed which

makes Python a very comprehensive and flexible tool. Additionally, Python is platform independent and runs on both Windows, Mac OS X and Linux/Unix systems. Various tools exist in order to wrap external source code and to make it accessible via the desired Python framework.

In the present work the Python package F2Py is used to wrap the several solvers presented in Chapter 2. The considered solvers are all written in the programming language Fortran, however, similar Python packages exist in order to wrap codes of other programming languages. An example is the package SWIG which can be used to wrap codes written in the programming language C. Wrapping the participating codes with F2Py or SWIG and thus making selected variables and routines accessible via Python is not only useful in terms of the desired coupling, it can also be useful in order to create a defined and controlled access for the end-user who can then employ the numerous opportunities of Python to conveniently manipulate or post-process the accessible data.

4.3 Wrapping and Coupling of Participating Models

This section gives an overview over how the coupling framework is built up and organized. It is shown how the source code of each single solver is wrapped into a shared object file and how those shared object files are then imported and accessed from Python in order to establish the desired coupling.

Figure 4.1 shows a schematic representation of the wrapping process. The example utilizes a source code written in Fortran and wrapped with F2Py, however, source codes written in other languages can be wrapped in a very similar way. In order to wrap the source code and thus making it accessible via Python the four following steps have to be carried out:

- Step 1:
  Define the *Standardized Interface Functions*
  - Analyse the execution process of the original stand-alone solver and redistribute the respective function calls into the available *Action Control Functions* listed in Section 4.4.1.
  - Define the *Data Flow Functions* listed in Section 4.4.2 needed to access

and manipulate selected variables of the stand-alone solver via the Python script.

- Step 2:
  Define the *Coupling Related Functions*
  - Define all additional functions needed to prepare the stand-alone solver for the coupled execution. The respective functions are dependent on the considered model but also on the other models that are coupled with it. More information can be found in Section 4.5.
  - Define all variables needed for the additional functionality of the coupling in a separate module.

- Step 3:
  Wrap everything into a shared object file

- Step 4:
  Use a Python script to import the Python Wrap and access the source code via the predefined standardized interface functions.

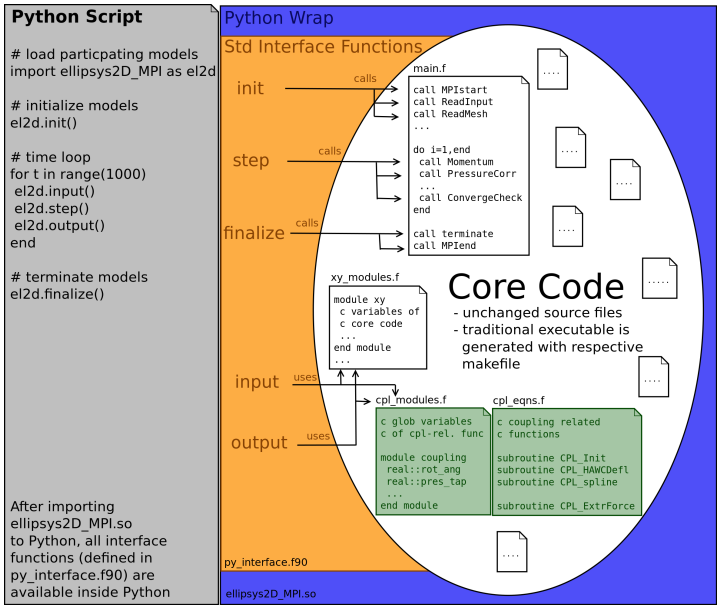


Figure 4.1: Schematic Representation of the Wrapping

The representation in Figure 4.1 tries to illustrate how the actual source code stays protected and untouched while the interface functions constitute the only access to the functionality of the stand-alone solver. Since the source code of

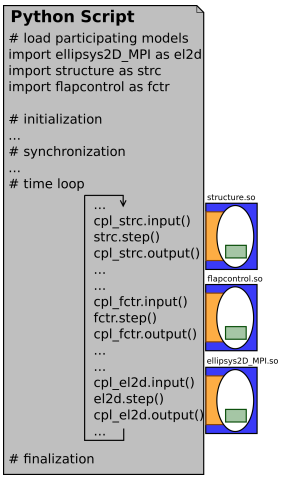


Figure 4.2: Schematic Representation of the Coupling

the stand-alone solver is not manipulated during the wrapping process the individual solvers can still be developed independently. However, in practice it is recommendable to consider at least the coupling related functions as a part of the source code and keep them under the same version control system. The coupling related functions are only activated if the solver is called from Python and runs in coupled mode, in all other cases a respective boolean variable is set to false and the functions are ignored.

After the Python wrap is imported into a Python script the predefined standardized interface functions can be used to control the execution of the embedded code. Importing several wraps into the same Python script allows to call the connected models alternately and to exchange the desired input and output data in between those calls. Figure 4.2 shows this basic idea of a Python coupling scheme. The given example refers to the aero-servo-elastic coupling between EllipSys2D, the 3-DOF structural code and the simple flap control algorithm presented in Section 2.3. However, the coupling between EllipSys3D and HAWC2 would look very similar or even simpler since in this case the control models are typically connected via the dll or shared object file interface of HAWC2 and not directly via the Python coupling framework.

The superordinate Python script controls and orchestrates the alternate execution of the different models, and since the framework has to manage the execution of computationally rather expensive and thus parallelized codes like the CFD solver EllipSys2D the framework has to be prepared for parallel computations as well.

An example of a typical parallel FSI computation is shown in Figure 4.3. The schematic representation illustrates that the Python script is executed on all participating computational nodes. The parallelized code EllipSys2D is then called on each node while the remaining serial codes are only called on the first node. The presented example again refers to the coupling of the models presented in Section 2.3, but the scheme would look very similar in case of a coupling between EllipSys3D and the serial structural solver HAWC2.

Figure 4.3 additionally visualizes the data flow related to a parallel execution of the present coupling scheme. While the solid arrows represent the internal data flow within the CFD code EllipSys2D the dashed arrows represent the data flow which has to be organized by the Python coupling framework. Potentially, each model communicates with each of the other connected models and running some models only on node 1 and other models on multiple nodes further complicates the data flow. Additionally, some variables only need to be exchanged during an initial synchronization step while other variables need to be exchanged at each time step. This rather complex data flow asks for a structured approach for the data flow handling. In Section 4.4.2 it will be explained how this structured approach looks like and how it was implemented in the present coupling framework.

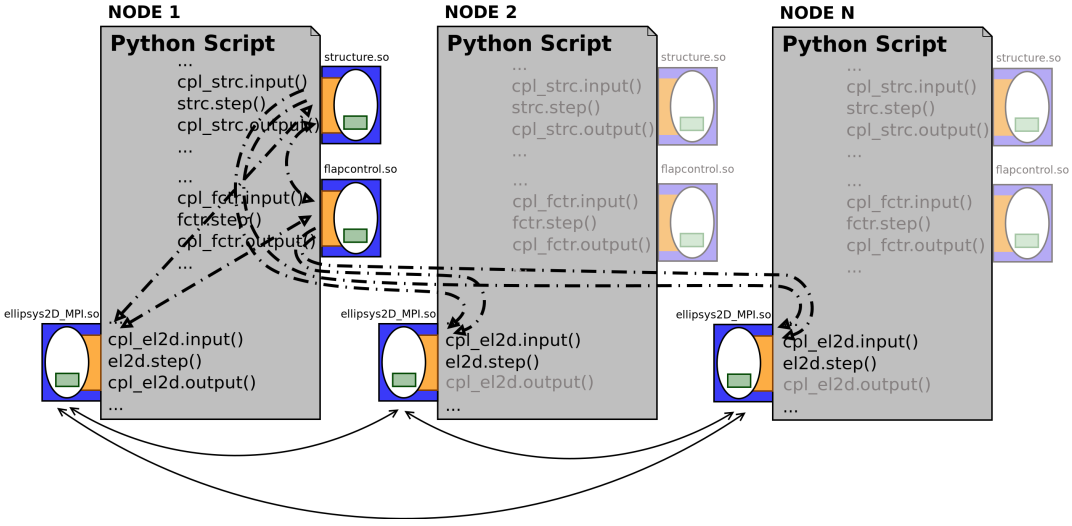


Figure 4.3: Schematic Representation of Coupling and Data Flow during Parallel Computation

4.4 Standardized Interface Functions

This section presents the implemented standardized interface functions which are called from the Python coupling framework in order to access the functionality of the connected models. A standardized way of accessing the models is necessary in order to provide a general coupling framework which allows to easily integrate a multitude of different codes in the future.

4.4.1 Action Control Functions

The action control functions are the interface functions needed to control the execution of the connected solvers and to advance the participating models in time. The execution process of every solver that is connected to the Python coupling scheme needs to be sorted into the following functions:

- INIT()
  The initialization step should include all routines needed by the participating solver in order to read in its eventual input files and to allocate and initialize the variables which solely depend on the information given within this solver.

- SYNC()
  The synchronization step is used to synchronize the participating solver with information from other participating solvers known from their individual initialization steps. Information like time step size, number of blades or the initial deflection state is communicated here and used to allocate and initialize the respective variables.

  - SYN1(), SYN2(), ..
    Additional synchronization steps can be used if more complex dependencies between the solvers exist (optional).

- STEP()
  In a coupled simulation the time loop of the stand-alone solver is moved to the superordinate Python script from where the function STEP() is called within each time iteration. The function includes all routines that are needed to advance the solver with one time step.

  For the present FSI simulation tool more sophisticated step functions are implemented as well:

  - STEP_PREDICT(), STEP_CORRECT()
    In order to establish a loose coupling as described in Section 3.1 the time step function of the structural solver has to be divided into a predictor and a corrector step.
  - STEP_INIT(), STEP_SUB(), STEP_FINALIZE()
    In order to establish a strong coupling as described in Section 3.1 both the outer time loop and the inner subiteration loop of the standalone solver have to be moved to the superordinate Python script. Within the subiterations the respective solver is advanced by calling the function STEP_SUB(). Outside the subiteration loop the time step starts with the function STEP_INIT() and concludes with the function STEP_FINALIZE().

- FINALIZE()
  This finalization step should include all functions needed to achieve a controlled termination of the solver.

4.4.2 Data Flow Functions

The data flow functions are used to send/receive the desired data to/from the models participating in the coupling. However, before presenting those functions on page 50 it first has to be explained how the data flow is generally organized within the Python coupling scheme.

A typical data flow of a coupled, parallel computation was already shown in Figure 4.3. Potentially, every model exchanges data with every other model and, depending on the connected models, some data has to be broadcast and some data has to be gathered from other computational nodes, some data has to be exchanged during every time step and some data has to be exchanged only once during the synchronization of the simulation. In order to simplify the organization of this rather complex data flow an algorithm is implemented which is based on the information given in a specific input file called *varlist.dat*. An abridgement of an exemplary input file is shown in Figure 4.4.

- The first column contains the string names of all the variables that have to be transferred during the coupling.

- The second column defines the data type of the exchange variable. The coupling scheme currently supports the following data types:

```
Variable     TYPE SOURCE RECV  t1   t2
 ...
'timestep'  RL1D  EL3D  HAWC INIT SYN1
'project'   CHAR  EL3D  HAWC INIT SYN1
'nblades'   ITGR  HAWC  EL3D INIT SYN1
'nsec'      IT1D  HAWC  EL3D INIT SYN1
'lEL3D'     BOOL  PYTH  HAWC INIT SYN1
'LHAWC'     BOOL  PYTH  EL3D INIT SYN1
'lLOOSE'    BOOL  PYTH  EL3D INIT SYN1
'lSTRONG'   BOOL  PYTH  EL3D INIT SYN1
 ...
'pitch'     RL1D  HAWC  EL3D SYN1 SYN2
'xini'      RL2D  HAWC  EL3D SYN1 SYN2
 ...
'pitch'     RL1D  HAWC  EL3D STEP STEP
'xtot'      RL2D  HAWC  EL3D STEP STEP
 ...
'lPRINTNOW' BOOL  PYTH  EL3D STEP STEP
'lPRINTNOW' BOOL  PYTH  HAWC STEP STEP
'lEL3D_END' BOOL  EL3D  PYTH STEP STEP
'lHAWC_END' BOOL  HAWC  PYTH STEP STEP
 ...
```

Figure 4.4: Abridgement of the Input File *varlist.dat* defining the Data Flow

  - BOOL: Boolean Data Type
  - CHAR: String Data Type
    (128 Characters)
  - ITGR, IT1D: Integer Data Type
    (Single Number or 1D Array of Arbitrary Length)
  - REAL, RL1D, RL2D, RL3D: Real Data Type
    (Single Number, 1D Array, 2D Matrix or 3D Matrix of Arbitrary Length)

- The third column of the input file tells the algorithm from which connected model the variable has to be read out. The fourth column tells to which model the variable has to be passed to. The four character strings of the currently available programs are:

  - EL2D, EL3D: EllipSys2D, EllipSys3D
  - STRC: Simple 3-DOF Structural Solver
  - HAWC: Aero-Elastic Code HAWC2
  - FCTR: Trailing Edge Flap Controller
  - PYTH: The Python Coupling Framework[1]

[1] Apart from the programs that are connected to the Python coupling framework, the framework itself can also be declared as the source or the receiver for a variable. This feature is utilized for variables declared inside Python and then passed to a participating model, or for variables that are fetched from a participating model and then used for post-processing inside the Python script.

- The fifth and sixth column are used to define the moment of the data exchange. The time instant $t1$ determines the moment of the variable read-out and the time instant $t2$ determines the moment of the variable read-in. The time instants $t1$ and $t2$ are defined by using the four character strings indicating the action control functions presented in Section 4.4.1.

The example file in Figure 4.4 includes some typical variables that are exchanged during the coupling between HAWC2 and EllipSys3D. For the purpose of a better illustration the variable list is split into four sections.

- The variables of the first section are exchanged between initialization and synchronization step. The variables are fetched after the participating models are initialized and their respective input files are read in. They are then transferred to their respective receivers and used to synchronize the connected models inside the subsequent synchronization step. Typically, the variables carry information about e.g. the chosen time step size, the number of blades and the number of utilized blade sections. At the same time some logical variables are transferred from the Python framework carrying information about the desired coupling scheme and information about which other models are connected. The latter is needed inside the participating solvers in order to activate the respective coupling related functions.

- Some synchronization variables cannot be set during a single synchronization step and a second synchronization step SYN2 is required. The necessity of an additional synchronization step is usually caused by the internal code structure of the connected stand-alone solvers. In order to comply with the minimal intrusion principle mentioned in Section 4.1 it is then preferred to rather add an additional synchronization step than introduce a severe change in the source code of the connected solver.

- The third section of variables includes the variables that need to be exchanged at every time step. This is indicated by setting both time instants $t1$ and $t2$ to 'STEP'. Typical variables exchanged in this block are the variables that describe the actual structural deflections and the actual aerodynamic forces.

- Some additionally variables are defined inside Python and sent to the connected models in order to trigger a synchronized force and deflection printout. Some other variables are transferred to Python in order to indicate

that individual convergence limits are met or that particular events demand for the termination of the coupled simulation.

After defining all necessary data in the input file *varlist.dat*, the data flow algorithm is supplied with all necessary information to organize the data flow in an automated way. This also includes the automatic broadcasting of variables in case data has to be transferred from a serial code running on node 1 and a parallel code running on all nodes.

The above mentioned structure and organization of the data flow is entirely managed inside the Python coupling framework and does not affect the connected models at all. The standardized interface functions which finally constitute the interface between the Python coupling framework and the participating solvers deal exclusively with the read-out and the read-in of the desired variables. For each supported data type listed on page 47 the following four data flow functions are implemented:

- PREP_INPUT_<DATATYPE>(*varname,dim*)
  This standardized interface function is used to allocate the solver variables inside the participating solver and to prepare the solver for the data input. The function is only called once at the beginning of the simulation.

- INPUT_<DATATYPE>(*varname,dim,value*)
  This standardized interface function is called to actually update the solver variable with the value given from Python.

- PREP_OUTPUT_<DATATYPE>(*varname,dim*)
  This standardized interface function is used to determine the exact dimensions of the output variable and to transfer the information to the Python framework.

- OUTPUT_<DATATYPE>(*varname,dim,value*)
  This standardized interface function is called to read out the actual value of the variable whenever it is desired.

In Figure 4.4 it can be seen that all input and output variables are identified via a specific string name. The string name is used inside Python to conveniently pick/drop the variable values from/into a Python dictionary where the values can

be easily accessed by using the respective string name.[2] The string is also used as an argument for the data flow functions in order to hook the variables of the Python dictionary with the respective variable inside the connected solvers. For the Fortran coded solvers EllipSys3D and HAWC2 the *case* statement is used for a convenient conjunction between Python string and Fortran variable. As an example Listing 4.1 shows an abridgement of an interface function callable from Python.

Listing 4.1: Hooking Exchange Variables via String Name

```
subroutine input_rl2d(varname,vardim1,vardim2,varvalin)
c-------------------------------------------------------------------------------
c       2Dim real arrays which are received from python
c-------------------------------------------------------------------------------
        ...
        real(kind=idp),dimension(vardim1,vardim2),intent(in)::varvalin
c-------------------------------------------------------------------------------
        select case(varname)
         case('xini')
           xini  =  varvalin
         case('xtot')
           xtot  =  varvalin
        ...
        end select
c-------------------------------------------------------------------------------
end subroutine input_rl2d
```

4.5 Coupling Related Functions

When a former stand-alone solver is executed in the coupled mode the solver needs to be equipped with additional functionality in order to comply with the new demands arising from the coupling. This additional functionality is very specific to the desired coupling. It is depending on the solver itself but also on the other solvers connected to it. In contrast to the standardized interface functions of the previous section the coupling related functions thus need to be discussed specifically for the respective coupling.

In Section 5.2 the most important and fundamental coupling related functions for the coupling of HAWC2 and EllipSys3D will be explained in more detail. In Section 6.1 the coupling related functions that are needed to couple the remaining other models will be discussed.

Remark:
Some more practical information about the developed Python coupling framework can be found in Appendix A. The section comprises information about the

[2] The Python dictionary not only contains the actual variable value but also all other variable specific information such as array dimension, exchange time and variable type

chosen file and code structure and gives a good idea about the general usage and the potentials of the framework.

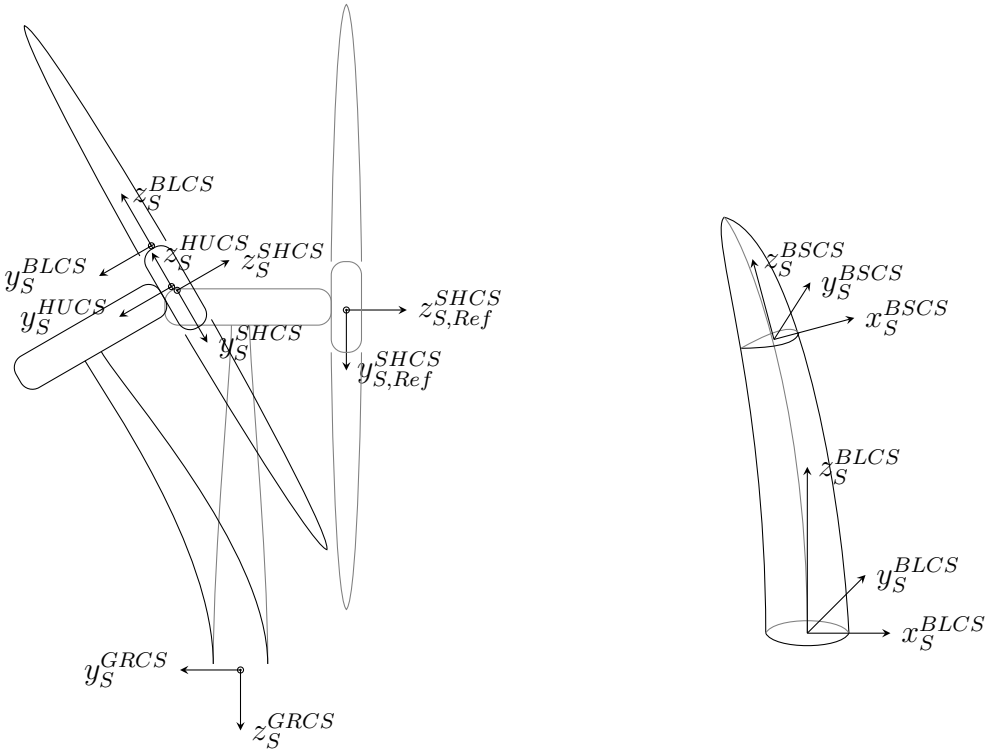
Chapter 5

Coupling between HAWC2 and EllipSys3D

The main focus of the present work is put on the FSI coupling between the multi-body structural solver of HAWC2 and the high-fidelity CFD solver EllipSys3D. The present chapter intends to explain how this coupling is realized in practice. After defining several coordinate systems and angles that are needed to conveniently describe the aerodynamic forces and the structural deflections during a coupled simulation in Section 5.1, it is explained in Section 5.2 which coupling related functions are implemented in order to facilitate the desired force and deflection transfer between the participating models. Section 5.3 is then discussing some specific issues of the implemented force and deflection transfer with respect to the guidelines of Section 3.2 proposed in order to establish a loosely coupled FSI simulation of sufficient time accuracy and sufficient numerical stability. After considering these specific issues together with the more general observations discussed in Chapter 3 it was decided to implement and test five different coupling schemes for the FSI coupling between HAWC2 and EllipSys3D. The respective coupling schemes are described in Section 5.4.

5.1 Coordinate Systems and Angle Definitions

The FSI coupling between HAWC2 and EllipSys3D has the aim of being minimally intrusive to the former stand-alone solvers. Therefore the coordinate systems and typical definitions of each solver are left unchanged. On the one hand this avoids changes inside the source code of the participating solvers, on the other hand, since all models are still set up in exactly the same way, it allows the



(a) Entire Wind Turbine (b) Turbine Blade

Figure 5.1: Coordinate Systems in HAWC2

usage of exactly the same input files as in the stand-alone computations. Keeping for each solver the original coordinate systems and definitions, however, results in certain conflicts at the coupling interface. These conflicts have to be resolved when transferring certain quantities from one model to the other. It is the task of the coupling framework to interchange the quantities appropriately.

In this section we present the most important coordinate systems needed for the coupling between HAWC2 and EllipSys3D. In the remainder of this work those coordinate systems will also be used to describe and analyse the deflections and forces of the FSI simulations. All presented coordinate systems are right-handed Cartesian coordinate systems.

Figure 5.1 shows the coordinate systems utilized in the structural solver HAWC2 assuming that the wind turbine model is built up with the same components as discussed in Section 2.1. In Figure 5.2 the respective coordinate systems are then utilized to define several typical angles needed to describe the deflection state of a wind turbine. All coordinate systems of the structural solver hold the letter S

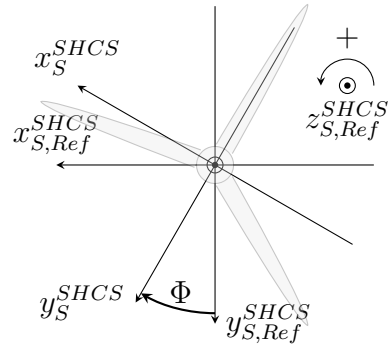
as subscript. The given coordinate systems are:

- Ground Coordinate System of HAWC2 ($GRCS_S$):
  - Origin at the root of the tower, the $z-axis$ points towards the ground, the $y-axis$ points along the main wind direction.
  - Base coordinate system of HAWC2
- Shaft Coordinate System of HAWC2 ($SHCS_S$):
  - Origin at the end of the rotor shaft i.e. in the centre of the rotor, the $z-axis$ points away from the shaft, the $y-axis$ points towards the ground in case the structure is stiff and the azimuthal angle is zero.
  - Used to determine the orientation of the hub centre.
- Shaft Reference Coordinate System of HAWC2 ($SHCS_S^{Ref}$):
  - Origin and axis like $SHCS_S$, but $SHCS_S^{Ref}$ is kept frozen at the initial position of the undeformed turbine model.
  - Misalignment with $SHCS_S$ gives information about the tilt angle α_t (Figure 5.2b), the yaw angle α_y (Figure 5.2c) and the azimuthal angle Φ (Figure 5.2a) which also includes the roll angle of the nacelle.
- Hub Coordinate System of HAWC2 ($HUCS_S$):
  - Origin at the root of the hub element, the $z-axis$ points towards the blade tip, the $y-axis$ points along the rotor shaft in case the structure is stiff and the cone angle is zero.
  - Misalignment with $SHCS_S$ gives information about the cone angle α_c (Figure 5.2d).
- Blade Coordinate System of HAWC2 ($BLCS_S$):
  - Origin at the midpoint of the circular root section, the $z-axis$ is perpendicular to the root section and points towards the blade tip, the $y-axis$ points along the rotor shaft in case the structure is stiff and both the cone and the pitch angle are zero.
  - Used to describe the half chord positions of the blade sections, misalignment with $HUCS_S$ gives information about the pitch angle Θ (Figure 5.2e).

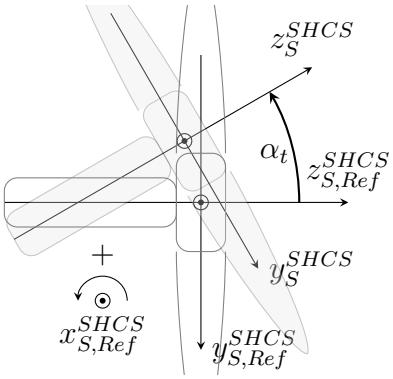
- Blade Section Coordinate System of HAWC2 ($BSCS_S$):
  - Origin at the half chord point of the respective blade section, the $x-axis$ is aligned with the chord line and points towards the leading edge, the $z-axis$ is perpendicular to the blade section and points towards the blade tip.
  - Misalignment with $BLCS_S$ gives information about the orientation of the blade section, including the twist angle ϵ (Figure 5.2f).

In Figure 5.3 the coordinate systems utilized in EllipSys3D are shown. While the base coordinate system of HAWC2 is located at the tower bottom of the wind turbine the CFD solver EllipSys3D defines its global coordinate system at the rotor centre of the undeformed rotor mesh. All other coordinate systems are defined very similar to HAWC2. Their origins are coinciding and only the orientation of the axis is different. The coordinate systems of the fluid solver hold the letter F as subscript.

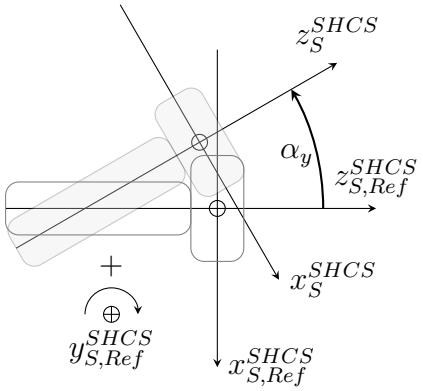
- Global Coordinate System of EllipSys3D ($GLCS_F$):
  - Origin at the rotor centre of the initial position of the undeformed turbine model, the $z-axis$ points along the main wind direction, the $y-axis$ points vertically upwards.
  - Base coordinate system of EllipSys3D, all grid point positions are stored with respect to this coordinate system.
- Shaft Coordinate System of EllipSys3D ($SHCS_F$):
  - Origin in the rotor centre, the $z-axis$ points along the rotor shaft, the $y-axis$ points vertically upwards in case the structure is stiff and the azimuthal angle is zero.
  - Counterpart to $SHCS_S$ in HAWC2
- Hub Coordinate System of EllipSys3D ($HUCS_F$):
  - Origin at the root of the hub, the $y-axis$ points towards the blade tip, the $z-axis$ points along the rotor shaft in case the structure is stiff and the cone angle is zero.
  - Counterpart to $HUCS_S$ in HAWC2
- Blade Coordinate System of EllipSys3D ($BLCS_F$):



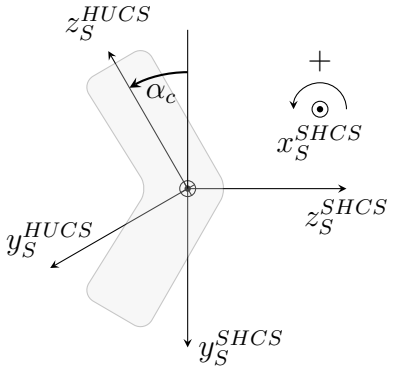
(a) Azimuthal Angle Φ (incl. Roll Angle)



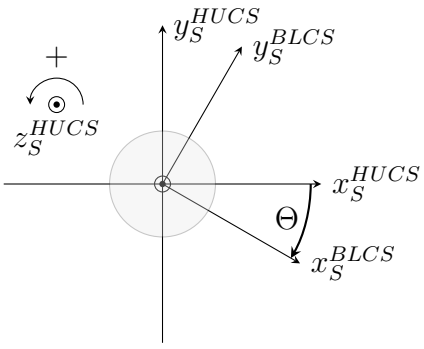
(b) Tilt Angle α_t



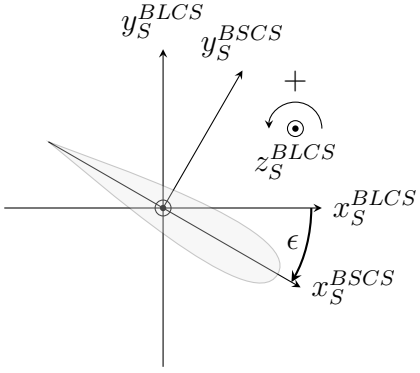
(c) Yaw Angle α_y



(d) Cone Angle α_c

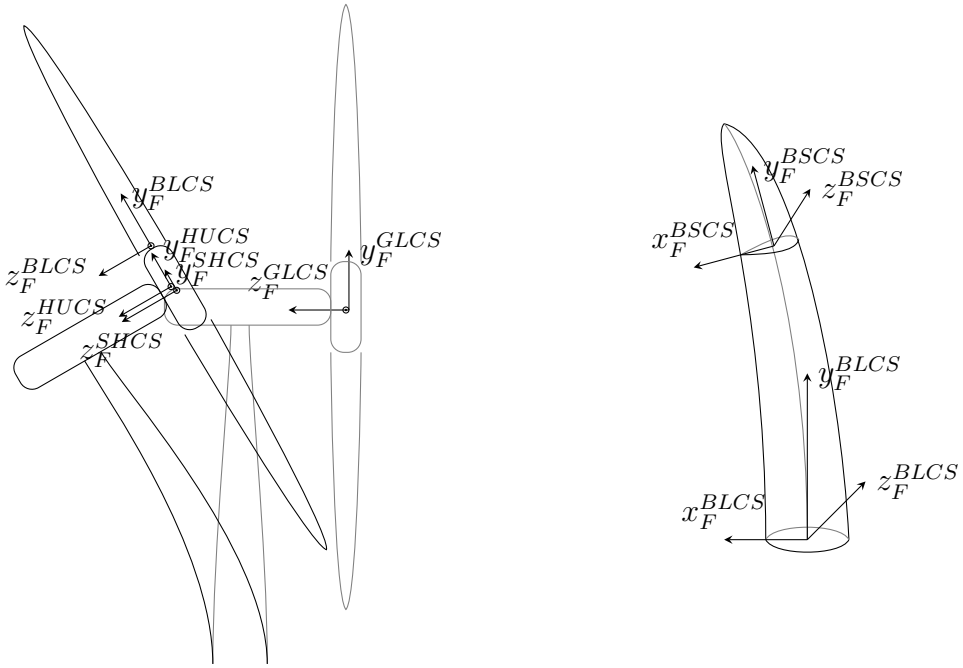


(e) Pitch Angle Θ



(f) Twist Angle ϵ

Figure 5.2: Angle Definitions using HAWC2 Coordinate Systems



(a) Entire Wind Turbine (b) Turbine Blade

Figure 5.3: Coordinate Systems in EllipSys3D

- Origin at the midpoint of the circular root section, the $y-axis$ is perpendicular to the root section and points towards the blade tip, the $z-axis$ points along the rotor shaft in case the structure is stiff and both the cone and the pitch angle are zero.
- Counterpart to $BLCS_S$ in HAWC2, the aerodynamic forces are read out in $BLCS_F$.

- Blade Section Coordinate System of EllipSys3D ($BSCS_F$):
  - Origin at the half chord point of the respective blade section, the $x-axis$ is aligned with the chord line and points towards the trailing edge, the $y-axis$ is perpendicular to the blade section and points towards the blade tip.
  - Counterpart to $BSCS_S$ in HAWC2

In order to transfer the forces and deflections at the domain interface between HAWC2 and EllipSys3D the conflicts due to the different coordinate system definitions have to be resolved appropriately. Details about the respective axis transformation are given in Appendix B.3.

5.2 Coupling Related Functions

This section presents the coupling related functions that are implemented in order to establish an appropriate force and deflection transfer between HAWC2 and EllipSys3D. Successively it will be explained how the structural deflections are read out in HAWC2, how the fluid mesh of EllipSys3D is deformed with respect to the given deflections, how the aerodynamic forces are read out in EllipSys3D and how these forces are applied on the structural model of HAWC2.

5.2.1 Reading Rotor Deflections in HAWC2

The definition of the rotor deflections is highly influenced by the organisation of the original stand-alone codes. In order to establish a minimally intrusive and relatively simple deflection transfer the goal was to define a convenient deflection read-out in HAWC2 which at the same time provides information that also can be used conveniently for the mesh deformation inside EllipSys3D.

Blade Deflection

In order to describe the deflections of the wind turbine blades it was decided to use the positions and orientations of the radially distributed aerodynamic sections illustrated in Figure 2.1. In the stand-alone version of HAWC2 the aerodynamic sections are used to determine and apply the aerodynamic forces from the BEM based aerodynamic model. However, it is also convenient to choose those sections for the definition of the actual blade deformation state as well. The respective information is easily retrievable from the HAWC2 source code and can also be used for the implemented force transfer explained in Section 5.2.3 and Section 5.2.4.

The position of each aerodynamic section is defined via its respective half chord point in $BLCS_S$ coordinates. The orientation of each aerodynamic section is then determined via the three *intrinsic Tait-Bryan angles* that are needed to distinctively describe the rotation from $BLCS_S$ coordinates to $BSCS_S$ coordinates. Similar to the well-known *Euler angles* the Tait-Bryan angles describe any rotation in 3D sufficiently. While the three Euler angles describe the rotation with the sequence $\alpha \rightarrow \beta \rightarrow \alpha$ and employ twice the same rotational axis, the Tait-Bryan angles describe the rotation with the sequence $\alpha \rightarrow \beta \rightarrow \gamma$ and thus use all three rotational axis. The sequence of the rotational axis can be chosen

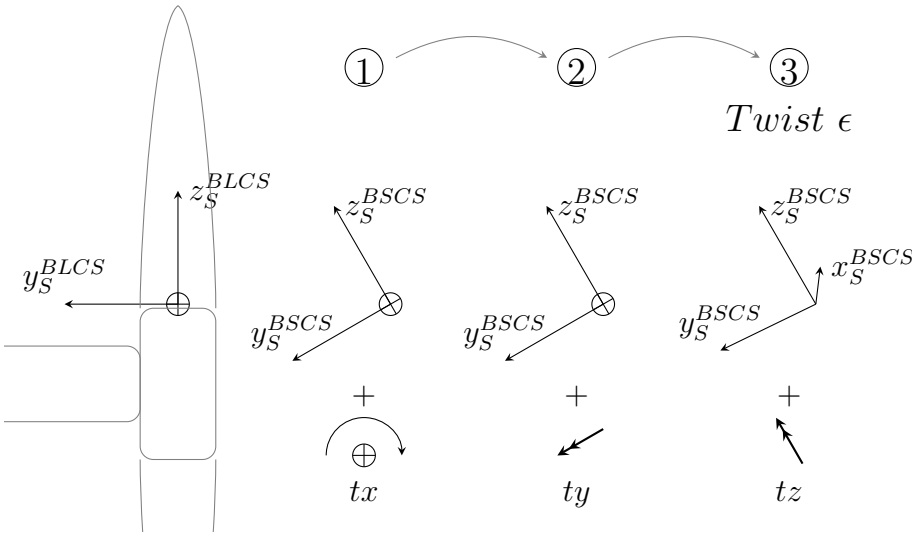


Figure 5.4: Intrinsic Tait-Bryan Rotations for an Aerodynamic Blade Section

freely but is then non-commutable, and rotating the axis in a different sequence would result in a different final rotation. The rotation sequence of the present work is chosen to be $tx \rightarrow ty \rightarrow tz$ where tx is the rotation around the $x - axis$, ty is the rotation around the $y - axis$ and tz is the rotation around the $z - axis$.

In Figure 5.4 it can be seen how this Tait-Bryan rotation sequence looks in terms of an aerodynamic blade section. Here, the angle tx corresponds to a rotation mainly caused by a flapwise bending of the blade, the angle ty corresponds to a rotation mainly caused by an edgewise bending, and the angle tz corresponds to a rotation indicating the blade twist angle ϵ. In order to obtain a simpler illustration the rotation around the $y - axis$ is set to zero. Since the rotational axis of the second and third rotation are not fixed but are moved by the previous rotations a so-called *intrinsic* Tait-Bryan formulation is used.

The position and orientation of each aerodynamic section is thus sufficiently defined by using the following six quantities

$$\mathbf{u}_{sec} = [x; y; z; tx; ty; tz]$$

where x, y, z are the half chord point coordinates in $BLCS_S$ and tx, ty, tz are the three intrinsic Tait-Bryan angles describing the rotation from $BLCS_S$ to $BSCS_S$.

Cone and Pitch Angle

As indicated in Figure 5.2d and Figure 5.2e the cone angle α_c and the pitch angle Θ can be found by describing the rotation from $SHCS_S$ coordinates to $HUCS_S$ coordinates and the rotation from $HUCS_S$ coordinates to $BLCS_S$ coordinates respectively. In both cases the expected rotation is mainly limited to one axis and one Tait-Bryan angle. The rotation from $SHCS_S$ coordinates to $HUCS_S$ coordinates is mainly limited to the angle tx (i.e. the cone angle α_c) and the rotation from $HUCS_S$ coordinates to $BLCS_S$ coordinates is mainly limited to the angle tz (i.e. the pitch angle Θ). Compared to these rather big rotations for coning and pitching the remaining rotations due to the elastic deformation are rather small and negligible. The cone and pitch angles are stored in the vectors $\mathbf{u}_{pitch}$ and $\mathbf{u}_{cone}$. In case a three bladed wind turbine is considered they write

$$\mathbf{u}_{pitch} = [\Theta^1; \Theta^2; \Theta^3]; \quad \mathbf{u}_{cone} = [\alpha_c^1; \alpha_c^2; \alpha_c^3]$$

Hubmotion

The position of the hub is described by the position of the rotor centre and the orientation of the shaft at that point. While the position is given by the x, y and z values in $GRCS_S$ coordinates, the orientation of the shaft is found by describing the rotation from $SHCS_S^{Ref}$ coordinates to $SHCS_S$ coordinates. Figure 5.5 shows the three respective intrinsic Tait-Bryan angles when using the chosen sequence of $tx \rightarrow ty \rightarrow tz$. The angle ty (i.e. the yaw angle α_y) is chosen to be zero in order to obtain a simpler illustration. It can be seen that the Tait-Bryan angle tx corresponds to a tilt angle α_t and the angle tz corresponds to the azimuthal angle Φ which also includes a possible roll angle α_r of the nacelle. The following six quantities are thus sufficient to describe the hub motion.

$$\mathbf{u}_{hub} = [x; y; z; tx; ty; tz]$$

Remark:
The misalignments between the several coordinate systems and thus the orientations of e.g. the aerodynamic sections or the hub centre can be also described by a respective 3x3 rotation matrix. These rotation matrices are accessible inside HAWC2 and extensively used there. However, the rotation matrices do not provide easily interpretable quantities and it was decided to use the above described

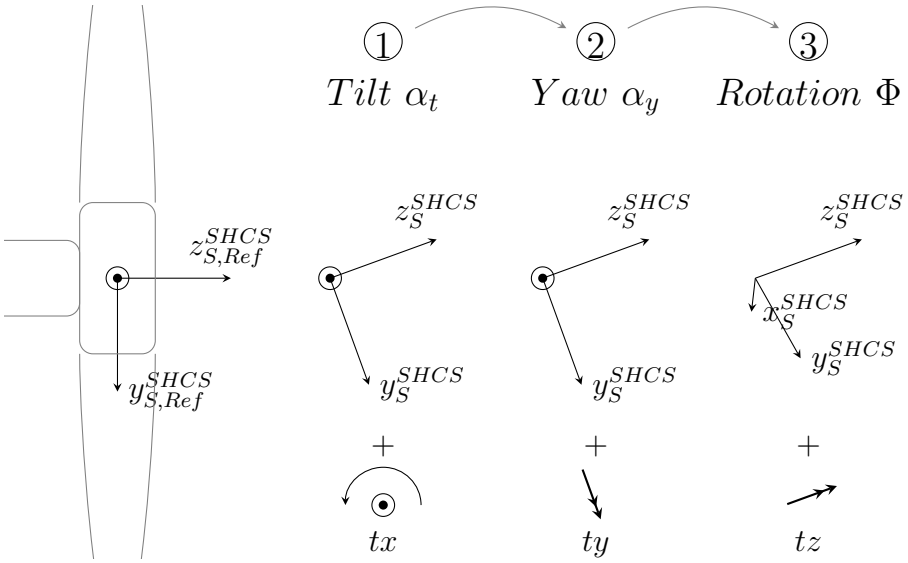


Figure 5.5: Intrinsic Tait-Bryan Rotations of the Hub

intrinsic Tait-Bryan angles instead. Inside HAWC2 a coupling related function is implemented which takes an arbitrary rotation matrix as input and outputs the desired intrinsic Tait-Bryan angles with the rotation sequence of $tx \rightarrow ty \rightarrow tz$.

5.2.2 Deforming Rotor Mesh in EllipSys3D

In the following it is explained how the rotor deflections of the previous section are used to deform the CFD mesh of the fluid solver. The moving mesh routines of the stand-alone solver EllipSys3D begin each new time step with the initial mesh positions $\mathbf{x}_{ini}$. Those initial mesh positions are loaded during the initialization step of the solver and typically include several initial deflections due to e.g. pre-bended, pre-twisted or swept blades. In order to find the actual positions $\mathbf{x}$ of the CFD rotor mesh at a certain time instant t we are thus only interested in the additional or effective deflections $\mathbf{x}_{def}$ which can be written as

$$\mathbf{x}_{def} = \mathbf{x} - \mathbf{x}_{ini}, \quad with \quad \mathbf{x} = \mathbf{x}_{ini} \; at \; t = 0 \tag{5.1}$$

In the same manner we also separate the structural deflections read out from HAWC2. In correspondence to equation 5.1 we can write

$$\mathbf{u}_{def} = \mathbf{u} - \mathbf{u}_{ini}, \quad with \quad \mathbf{u} = \mathbf{u}_{ini} \; at \; t = 0 \tag{5.2}$$

where the vector **u** contains all structural deflections introduced in the previous Section:

$$\mathbf{u} = [\mathbf{u}_{sec1}; \mathbf{u}_{sec2}; ..; \mathbf{u}_{pitch}; \mathbf{u}_{cone}; \mathbf{u}_{hub}] \tag{5.3}$$

For a later use an additional vector **ub** is defined which in contrast to **u** excludes the hub motion $\mathbf{u}_{hub}$ and only includes the deflections related to the blade deformations:

$$\mathbf{ub} = [\mathbf{u}_{sec1}; \mathbf{u}_{sec2}; ..; \mathbf{u}_{pitch}; \mathbf{u}_{cone}] \tag{5.4}$$

Accordingly, the vector **uh** is defined with

$$\mathbf{uh} = [\mathbf{u}_{hub}] \tag{5.5}$$

The rotor deformation is then carried out in the following steps:

1. Commencing with the initial rotor mesh the rotor deformation starts by turning the mesh until the blade coordinate system $BLCS_F$ of blade b is aligned with the global coordinate system $GLCS_F$.

2. Since the vectors $\mathbf{u}_{sec}$ are only known at some specific radial blade positions but need to be known for each grid point along the blade its values are spline interpolated with respect to the radial positions of each mesh vertex. In this case the radial blade position is the y coordinate of the $BLCS_F$ coordinate system.

3. Each mesh vertex is now rotated around the respective half chord point using the three intrinsic Tait-Bryan angles tx, ty and tz. As seen in Figure 5.4 an intrinsic rotation demands the rotational axis to be moved with the precedent rotations, however, with a small modification it is also possible to use the fixed axis of the blade coordinate system $BLCS_F$ in order to conduct the desired intrinsic rotation. This is done by reversing the original rotation sequence and using the fixed axis of the $BLCS_F$ coordinate system together with the sequence $tz \rightarrow ty \rightarrow tx$ instead.

4. After the rotation is completed each mesh vertex is translated corresponding to the interpolated values using the first three entries of the vectors $\mathbf{u}_{sec}$.

5. Finally the pitch angle and then the cone angle of blade b are introduced by rotating all vertices around the respective axis. The angles are found inside the vectors $\mathbf{u}_{pitch}$ and $\mathbf{u}_{cone}$.

   Remark:
   Carrying out Step 2 to Step 5 alone would not lead to a useful new mesh configuration. So far every mesh vertex is translated and rotated in correspondence to the spline interpolation of the vectors $\mathbf{u}_{sec}$ which is only based on the radial position of the respective vertices. With an increased normal distance to the blade the rotations tx, ty and tz around the half chord point would lead to relatively large vertex movements and finally to intersecting grid lines. Additionally, the mesh motions would interfere with the mesh motions of the neighbouring blades. However, with regard to a better analytical treatment in the remainder of this work we interpret those first steps of the mesh deformation as a transformation of the form:

$$\mathbf{ub}_{def} \mapsto \mathbf{xb}_{def} \tag{5.6}$$

   which indicates that the structural deflection vector $\mathbf{ub}_{def}$, using the discretization of the structural solver, transforms to the new structural deflection vector $\mathbf{xb}_{def}$, using the discretization of the fluid solver.

6. As mentioned in the previous step the motion of the vertices has to be limited to the area near the blade in order to smear out the mesh deformation towards the farfield and in order to not interfere with the mesh motion of the neighbouring blade. This is done by using a blendfactor F which has the value of $F = 1$ on the blade surface and, after a smooth transition, approaches the value of $F = 0$ at a certain distance away from the blade. The blendfactors are based on the initial positions of the mesh vertices and stay constant during the entire simulation. As an example a visualization of the blendfactors around a blade of the NREL 5MW reference turbine is given in Figure 7.1a.
   Gathering the blendfactors of all mesh vertices in the vector $\mathbf{F}$ and utilizing the vector $\mathbf{xb}_{def}$ from equation 5.6, the determination of the actual mesh

position can be written in the form:

$$\mathbf{x} = \mathbf{F}^b \cdot (\mathbf{x}_{ini} + \mathbf{xb}^b_{def}) + (\mathbf{F}^b - 1) \cdot \mathbf{x}_{ini} \tag{5.7}$$

where the superscript b indicates that we are considering the deformation of blade b only.

7. After all blades have been successively deformed in the above explained manner (Step 1 to Step 6) the full rotor is rotated using the last three entries of $\mathbf{uh}_{def}$. The rotation takes place around the hub centre which is initially located in the origin of the fixed coordinate system $GLCS_F$. In order to realize the intrinsic Tait-Bryan rotation by using the fixed axis of the $GLCS_F$ coordinate system we have to reverse the rotation sequence of Figure 5.5 and use the sequence $tz \rightarrow ty \rightarrow tx$ instead, i.e. the sequence $\Phi \rightarrow \alpha_y \rightarrow \alpha_t$.

8. The hub centre is finally translated using the first three entries of $\mathbf{uh}_{def}$. As done previously in Equation 5.6 we can interpret the previous two steps as a transformation of the structural deflection into the discretization of the fluid solver

$$\mathbf{uh}_{def} \mapsto \mathbf{xh}_{def} \tag{5.8}$$

Remark:
It should be mentioned that the mesh motion conducted in Step 7 and Step 8 does not deform the mesh, instead it conducts a rigid body motion where all grid cells maintain their prior geometry.

9. A superposition with the blade deformations calculated in Equation 5.7 is then giving the final mesh position of the actual time step.

$$\mathbf{x} = \mathbf{x} + \mathbf{xh}_{def} \tag{5.9}$$

5.2.3 Applying Aerodynamic Forces in HAWC2

It was already mentioned before that the stand-alone version of HAWC2 determines and applies the aerodynamic forces at the locations of the radially distributed aerodynamic sections illustrated in Figure 2.1. In order to keep the

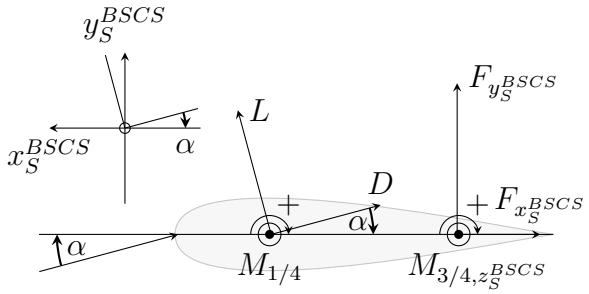
coupling related functions minimal intrusive it was decided to maintain those aerodynamic sections also during the coupling with EllipSys3D as the positions where the aerodynamic forces are applied to the structure.

In the stand-alone version of HAWC2 the lift force L, the drag force D and the moment around the quarter chord point $M_{1/4}$ are found from the tabulated values of two-dimensional aerofoil data and are then applied at the three-quarter chord point of the respective aerodynamic section using $BSCS_S$ coordinates. In Figure 5.6a it is illustrated how the force components L and D are transformed to the quantities $F_{x_S^{BSCS}}$ and $F_{y_S^{BSCS}}$ and how the moment $M_{1/4}$ is moved to the three quarter chord value $M_{3/4,z_S^{BSCS}}$.

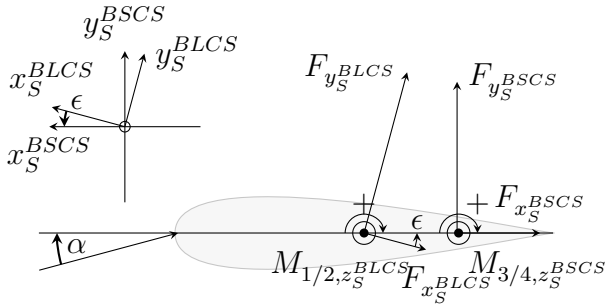
In the coupled mode the aerodynamic forces are provided from the CFD solver EllipSys3D. The respective coupling related functions read out the forces in $BLCS_F$ coordinates and determine the aerodynamic moments around the sectional half chord points (see Section 5.2.4). HAWC2 receives those forces in $BLCS_S$ coordinates and applies them on the three-quarter chord point of the aerodynamic section using $BSCS_S$ coordinates. In Figure 5.6b it is illustrated how the force components $F_{x_S^{BLCS}}$ and $F_{y_S^{BLCS}}$ are transformed to the components $F_{x_S^{BSCS}}$ and $F_{y_S^{BSCS}}$ and how the moment $M_{1/2,z_S^{BLCS}}$ is transformed and moved to the moment $M_{3/4,z_S^{BSCS}}$.

In order to achieve a simplified illustration in Figure 5.6b the chosen example only considers the presence of a certain twist angle ϵ and assumes the remaining Tait-Bryan angles tx and ty to be equal to zero. However, in reality the rotation from $BLCS_S$ coordinates to $BSCS_S$ coordinates demands indeed for a three-dimensional transformation using all three Tait-Bryan angles.

The aerodynamic forces are applied in exactly the same positions and in exactly the same units as before in the stand-alone version of HAWC2. The lift and drag forces have the unit [N/m] and the moments have the unit [Nm/m]. This means that all subsequent routines of HAWC2 can be left unchanged and that the implemented coupling related functions indeed comply to the requirements of being minimal intrusive.



(a) Using Traditional BEM Forces



(b) Using Forces from EllipSys3D

Figure 5.6: Applying Aerodynamic Forces in HAWC2

5.2.4 Extracting Aerodynamic Forces in EllipSys3D

In the previous Section it was mentioned that the aerodynamic forces have to be known at the radially distributed aerodynamic sections. In terms of a respective force read-out in EllipSys3D this is a very convenient circumstance since the positions and orientations of the aerodynamic sections are already used to describe the blade deflections of Section 5.2.1. There is thus no need to transfer any additional information from HAWC2 to EllipSys3D in order to conduct an appropriate force read-out at the respective blade positions.

In order to provide the aerodynamic forces in the desired unit of [N/m] the forces are extracted from the fluid mesh of EllipSys3D by computing a line integral along the respective aerofoil sections. The resulting moments with the unit [Nm/m] are conveniently determined around the half chord points of each section since those coordinates are directly available from the structural information received from HAWC2. The forces and moments are determined in $BLCS_F$ coordinates.

Unfortunately, the current implementation of the force read-out does not account for the actual sectional orientation. As indicated in Figure 5.7 the cutting

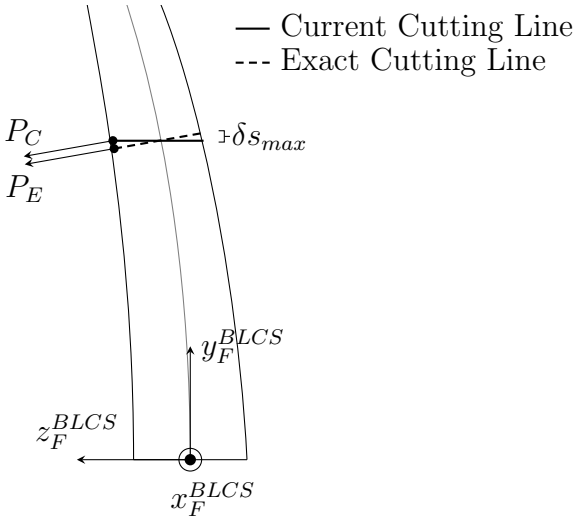


Figure 5.7: Cutting Lines and Pressure Force Orientation during Force Read-Out in EllipSys3D

line of the line integral is instead kept parallel to the $xz-plane$ of the $BLCS_F$ coordinate system. This is due to the currently employed routine that computes the line integral. The routine can only accomplish horizontal cuts and would require relatively expensive coordinate transformations in order to achieve an exact positioning of each respective cutting line. However, it is not expected that a slightly displaced cutting line results in a noticeably erroneous force read-out since the currently extracted pressure forces P_C are still read out perpendicular to the blade surface and have thus the same orientation as the desired exact pressure forces P_E. As illustrated in Figure 5.7 the only difference in the determination of the pressure forces lies in the slightly inaccurate blade position for the read-out. The maximum deviation in the blade position is indicated with δs_{max}.

Since a CFD simulation is able to catch the essential three-dimensional effects of a wind turbine rotor the extracted aerodynamic forces from EllipSys3D are considered to be highly three-dimensional. This is in contrast to the BEM based aerodynamic model of a traditional HAWC2 simulation which computes the aerodynamic forces from two-dimensional aerofoil data only. However, after the aerodynamic forces of EllipSys3D are read out in $BLCS_F$ coordinates and then transformed to $BSCS_S$ coordinates the major force components are still lying in the $xy-plane$ of the $BSCS_S$ coordinate system. The contributions of the three out-of-plane components $F_{z_S^{BSCS}}$, $M_{x_S^{BSCS}}$ and $M_{y_S^{BSCS}}$ are rather small and stem from the spanwise surface friction and eventual inaccuracies during the force read-out and the preceding force transformation.

5.3 Specific Issues of the Coupling

5.3.1 Non-Conservative and Conservative Force and Deflection Transfer

The implemented force and deflection transfer as explained in Section 5.2 is not energy conservative. The three integrated sectional force components lift, drag and moment are extracted from the fluid mesh at certain radially distributed blade positions and are then transferred to the structural model of HAWC2 where they are assumed to vary linearly in-between the given points. As indicated in Figure 5.8 it is due to the different spatial discretizations of fluid and structural mesh that this force transfer does not lead to an equilibrium between the energy released by the fluid and the energy received by the structure. The chosen non-conservative force transfer can thus be compared with the non-conservative approach discussed in Section 3.2.7 and illustrated in Figure 3.3.

In Farhat [6] such a non-conservative approach is denoted as a *consistent* approach. Several FSI couplings choose a consistent approach rather than a conservative approach. In the work of Farhat it is mentioned that a consistent approach can also lead to reliable, accurate and numerically conservative results. In the work of De Boer [73], [74] it is stated that a consistent approach has no effect on the accuracy and stability of the FSI computation in case a loosely coupled scheme is employed where the inherent energy inaccuracy is bigger than the erroneous energy introduced by the consistent load transfer. It is also mentioned that choosing a conservative approach can lead to non-physical oscillations in the pressure forces.

A non-conservative force and deflection transfer was implemented in the present work since the structural solver of HAWC2 still receives the aerodynamic forces in exactly the same way as in the traditional stand-alone version. The implementation thus complies to the demands of establishing a minimally intrusive coupling as formulated in Section 4.1. Additionally, the chosen non-conservative approach only communicates descriptive and easily interpretable quantities which makes it possible to easily monitor and control the data transfer between the coupled models.

Observing Figure 5.8 it can be easily seen that the erroneous energy introduced

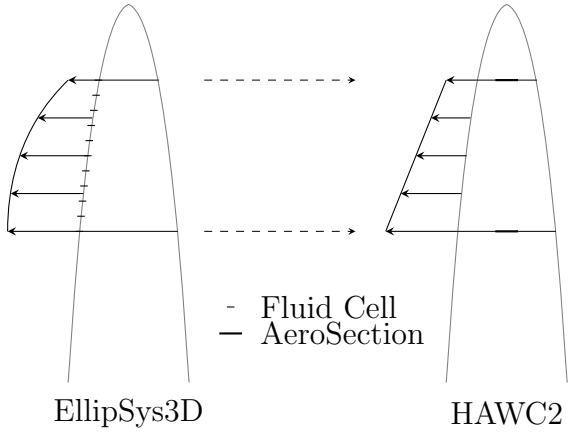


Figure 5.8: Non-Conservative Load Transfer between EllipSys3D and HAWC2

by the implemented non-conservative load transfer can be minimized by increasing the number of aerodynamic sections. However, due to the different spatial discretizations of fluid and structure this approach will neither converge to a fully energy conservative load transfer. In order to better investigate the influences of this non-conservative load and deflection transfer it was thus decided to define and monitor several energies related to the process. In case a loose coupling scheme is utilized the non-conservative load transfer at the FS interface can be assessed by observing the energies

$$\Delta {E_{F,FS}}^* = (\frac{\mathbf{F}_F^{n+1} + \mathbf{F}_F^n}{2}) \cdot (\mathbf{x}_\Gamma^{n+1^*} - \mathbf{x}_\Gamma^{n^*}) \tag{5.10}$$

and

$$\Delta {E_{S,FS}}^* = (\frac{\mathbf{F}_S^{n+1} + \mathbf{F}_S^n}{2}) \cdot (\mathbf{u}^{n+1^*} - \mathbf{u}^{n^*}) \tag{5.11}$$

where $\Delta {E_{F,FS}}^*$ represents the energy released by the fluid solver during the time interval $[t^n,t^{n+1}]$ and where $\Delta {E_{S,FS}}^*$ represents the energy received by the structure during the same time period. As indicated with the asterisk both energies are based on the predicted deflection state in order to be indeed comparable. In case a strong coupling scheme is utilized the non-conservative load transfer can be assessed by observing the respective energies

$$\Delta E_{F,FS} = (\frac{\mathbf{F}_F^{n+1} + \mathbf{F}_F^n}{2}) \cdot (\mathbf{x}_\Gamma^{n+1} - \mathbf{x}_\Gamma^n) \tag{5.12}$$

and

$$\Delta E_{S,FS} = (\frac{\mathbf{F}_S^{n+1} + \mathbf{F}_S^n}{2}) \cdot (\mathbf{u}^{n+1} - \mathbf{u}^n) \tag{5.13}$$

where the two energies are based on the updated deflection state of the actual subiteration.

A comparison between ${\Delta E_{F,FS}}^*$ and ${\Delta E_{S,FS}}^*$ as well as a respective comparison between $\Delta E_{F,FS}$ and $\Delta E_{S,FS}$ will provide information about the erroneous energy that is introduced by the implemented non-conservative load transfer. Furthermore, the extracted energies can be utilized to establish a simple force scaling that finally enforces a energy conservative load transfer. In terms of a loosely coupled algorithm this force scaling enforces the energy balance

$${\Delta E_{F,FS}}^* \doteq {\Delta E_{S,FS}}^* \tag{5.14}$$

by scaling the entire force vector $\mathbf{F}_S^{n+1}$ with

$$\mathbf{F}_S^{n+1} = \tilde{\mathbf{F}}_S^{n+1} \cdot \frac{{\Delta E_{F,FS}}^*}{{\Delta \tilde{E}_{S,FS}}^*} \tag{5.15}$$

where the tilde sign indicates the original values before the scaling. It is important to note that the claim of Equation 5.14 is different to the problem formulated in Equation 3.3. While Equation 3.3 can't (and shouldn't!) be fulfilled due to the different deflection states the energy calculations are based on, the energies of Equation 5.14 are based on the same predicted deflection states and are thus indeed comparable. Fulfilling Equation 5.14 ensures an energy conservative force transfer in terms of the spatial transformation, however, it has no influence on the inherent and persistent energy inaccuracy of a loose coupling scheme in general.

In terms of a strongly coupled algorithm the respective energy balance writes

$$\Delta E_{F,FS} \doteq \Delta E_{S,FS} \tag{5.16}$$

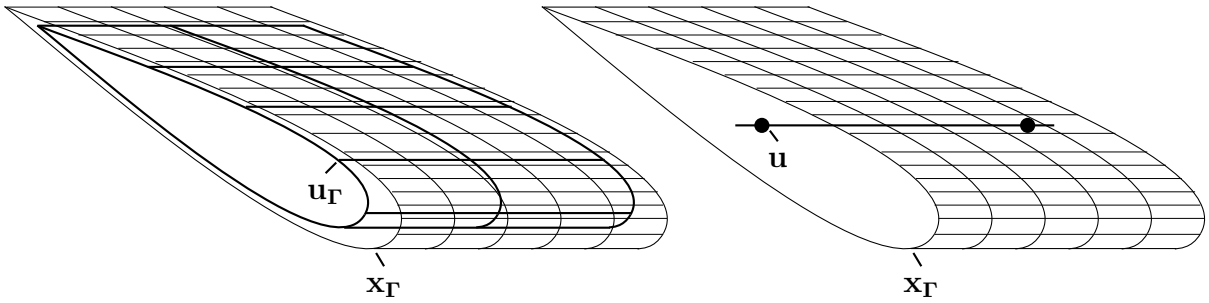
and is obtained with the force scaling

$$\mathbf{F}_S^{n+1} = \tilde{\mathbf{F}}_S^{n+1} \cdot \frac{\Delta E_{F,FS}}{\Delta \tilde{E}_{S,FS}.} \tag{5.17}$$

Section 5.4 lists all implemented coupling methods employed to couple HAWC2 with EllipSys3D. All coupling methods employ the standard non-conservative force and deflection transfer as explained in Section 5.2, however, in the coupling schemes of Section 5.4.2 and Section 5.4.5 the simple force scaling of Equation 5.15 and Equation 5.17 is employed in order to enforce an energy conservative force transfer. The explanations of the implemented coupling schemes in Section 5.4 are accompanied by specific flow charts which might further clarify the different attempts to optimize the FSI coupling in terms of energy accuracy and thus stability.

Remark 1: An energy conservative alternative to the implemented load transfer is outlined in Appendix B.4. The respective draft is based on the energy conservative load transfer discussed in Brown [72]. The draft can be compared with the approach described in Section 3.2.7 but adapts to the highly non-matching mesh configuration between HAWC2 and EllipSys3D as illustrated in Figure 5.9.

Remark 2: As illustrated in Figure 5.9b the coupling between HAWC2 and EllipSys3D exhibits a highly non-matching mesh configuration at the FS interface. In such a configuration the structural mesh does not share a common wet surface Γ with the fluid mesh. The structural deflections $\mathbf{u}$ used in the energy definitions of the present section are thus not indicated with the respective subscript Γ. This is different to the energy definitions of Section 3.2.1 where the general discussion assumes a more typical non-matching mesh configurations such as illustrated in Figure 5.9a.



(a) Typical Non-Matching Meshes (b) Highly Non-Matching Meshes

Figure 5.9: Interface between Fluid and Structure Mesh

5.3.2 Monitoring Energies during Deflection Transfer

By employing a conservative load and motion transfer as discussed in Section 3.2.7 and outlined in Appendix B.4 the deflection transfer is coupled to the load transfer using the same but transposed transfer matrix. In the implemented non-conservative approach, however, the load transfer is independent from the deflection transfer. Considering the deflection transfer as an independent event it could be interesting to observe the respective energies that are released by the structure and received by the fluid during that transfer. In a loose coupling those energies could be defined with

$$\Delta {E_{F,SF}}^* = \mathbf{F}_F^n \cdot (\mathbf{x}_\Gamma^{n+1^*} - \mathbf{x}_\Gamma^{n^*}) \tag{5.18}$$

and

$$\Delta {E_{S,SF}}^* = \mathbf{F}_S^n \cdot (\mathbf{u}^{n+1^*} - \mathbf{u}^{n^*}) \tag{5.19}$$

where the subscript SF indicates that the energies are connected to the deflection transfer from the structure S to the fluid F. Since the new force vector is not yet known during the deflection transfer both energies are entirely based on the force vectors of the actual time step. In a strong coupling the respective energies can be written as

$$\Delta E_{F,SF} = \mathbf{F}_F^n \cdot (\mathbf{x}_\Gamma^{n+1} - \mathbf{x}_\Gamma^n) \tag{5.20}$$

and

$$\Delta E_{S,SF} = \mathbf{F}_S^n \cdot (\mathbf{u}^{n+1} - \mathbf{u}^n). \tag{5.21}$$

Originally, it was thought to employ a deflection scaling that enforces $\Delta {E_{F,SF}}^* \doteq \Delta {E_{S,SF}}^*$ and $\Delta E_{F,SF} \doteq \Delta E_{S,SF}$ respectively and works in a similar way as the force scaling of Equation 5.15 and Equation 5.17. However, such a deflection scaling was not feasible since the implementation was triggering unstable computations. The respective energies are thus only monitored during the FSI simulations but are not used for any scaling. The energies can also be found in the flow charts of Section 5.4.

5.3.3 Time-Independent Mesh Deformation Matrix

In Section 3.2.3 it was pointed out that special care needs to be taken while choosing the mesh deformation matrix $\bar{\mathbf{U}}$. It was stated that the mean value of the time dependent matrix $\mathbf{U}(t)$ has to be determined in accordance to the chosen time integration scheme of the fluid solver. In the following it will be argued that the mesh deformation inside EllipSys3D is corresponding to a mesh deformation that employs a time independent mesh deformation matrix $\mathbf{U}(t) = const$. This will lead to the simple choice of $\bar{\mathbf{U}} = const$, a choice which is insensitive to the employed time integration scheme and which will thus not degrade the time accuracy of the coupled system.

As seen in Equation 3.7 the mesh deformation matrix is used to translate the structural deformation $\mathbf{u}_\Gamma$ of the wet surface Γ to all grid points $\mathbf{x}$ of the fluid mesh. However, as illustrated in Figure 5.9b, the highly non-matching mesh configuration between HAWC2 and EllipSys3D does not provide a corresponding wet surface.
In Section 5.2.2 it is described how the structural deflections of the one-dimensional beam elements $\mathbf{ub}_{def}$ are transformed to a corresponding deformation of the three-dimensional CFD mesh. The transformations related to Equation 5.6 define the deformation of the entire volume mesh $\mathbf{xb}_{def}$ and thus certainly also the deformation of the wet blade surface $\mathbf{xb}_{def,\Gamma}$. In order to formulate a mesh deformation matrix $\bar{\mathbf{U}}$ for the mesh deformation in EllipSys3D it is necessary to define an expression that connects the motion of the surface points $\mathbf{xb}_{def,\Gamma}$ with the respective vertex positions of the entire volume mesh $\mathbf{xb}_{def}$.

The mesh deformation related to Equation 5.6 is illustrated in Figure 5.10 where, for simplicity, the deformation is limited to the rotation tx of one certain blade section only. The figure shows how the rotation of the half chord line is extrapolated to the blade surface Γ in order to move the surface point P_Γ with the distance $\Delta y^R_{P_\Gamma}$. It can also be seen how the same rotation moves the volume mesh point P with the distance Δy^R_P. By employing the intercept theorem, which is valid for sufficiently small rotation angles, a certain relation can be defined between the motion of the two points, stating that point P moves with the distance $\Delta y^R_P = d_P/d_{P_\Gamma} \cdot \Delta y^R_{P_\Gamma}$. Considering all three rotations and translations involved in a blade deformation the relation between the surface point P_Γ and the point P located at a certain distance away from the blade can be expressed with

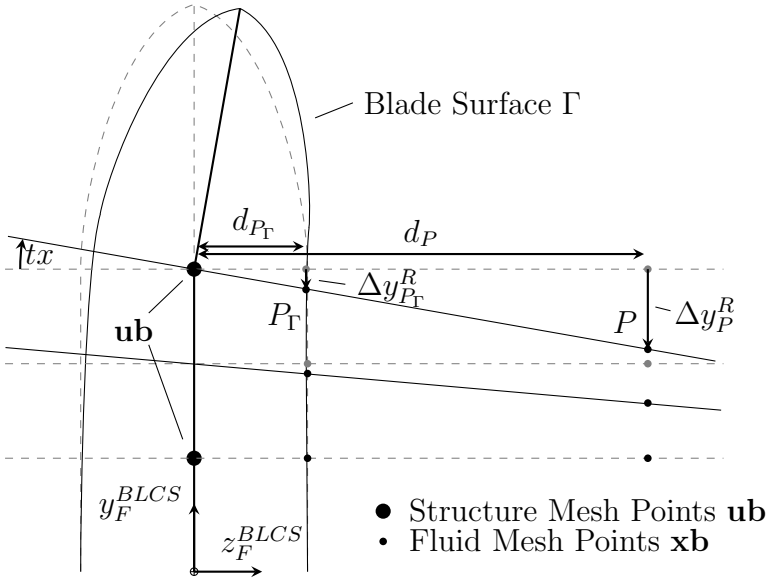


Figure 5.10: Mesh Deformation due to Blade Motion

$$\begin{bmatrix}\Delta x\\ \Delta y\\ \Delta z\end{bmatrix}_P = \underbrace{\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}}_{A^T} \cdot \begin{bmatrix}\Delta x^T\\ \Delta y^T\\ \Delta z^T\end{bmatrix}_{P_\Gamma} + \underbrace{\begin{bmatrix}\frac{d_P}{d_{P_\Gamma}} & 0 & 0\\ 0 & \frac{d_P}{d_{P_\Gamma}} & 0\\ 0 & 0 & \frac{d_P}{d_{P_\Gamma}}\end{bmatrix}}_{A^R} \cdot \begin{bmatrix}\Delta x^R\\ \Delta y^R\\ \Delta z^R\end{bmatrix}_{P_\Gamma} \tag{5.22}$$

where the time independent matrix $\mathbf{A}^T$ transforms the change in x, y and z due to the translational motion of P_Γ and the time independent matrix $\mathbf{A}^R$ transforms the change in x, y and z due to the rotational motion of P_Γ. Joining $\mathbf{A}^T$ and $\mathbf{A}^R$ to the matrix $\mathbf{A}$ and using an appropriate representation for all surface points results in the equation

$$\mathbf{xb}_{def} = \mathbf{A} \cdot \mathbf{xb}_{def,\Gamma} \tag{5.23}$$

The formulation of Equation 5.23 can now be used in Equation 5.7 in order to obtain the following expression for the grid points of the new time step $n+1$

$$\mathbf{x}^{n+1} = \mathbf{F} \cdot (\mathbf{x}_{ini} + \mathbf{A} \cdot \mathbf{xb}^{n+1}_{def,\Gamma}) + (\mathbf{F} - 1) \cdot \mathbf{x}_{ini} \tag{5.24}$$

After calculating $\Delta x = \mathbf{x}^{n+1} - \mathbf{x}^n$ with

$$\mathbf{x}^{n+1} - \mathbf{x}^{n} = \mathbf{F} \cdot \mathbf{A} \cdot (\mathbf{xb}^{n+1}_{def,\Gamma} - \mathbf{xb}^{n}_{def,\Gamma}) \tag{5.25}$$

and moving $\mathbf{x}^n$ to the right side of the equation

$$\mathbf{x}^{n+1} = \mathbf{x}^{n} + \mathbf{F} \cdot \mathbf{A} \cdot (\mathbf{xb}^{n+1}_{def,\Gamma} - \mathbf{xb}^{n}_{def,\Gamma}) \tag{5.26}$$

we can easily compare the result with Equation 3.7 and see that the implemented mesh deformation of the considered FSI coupling between HAWC2 and EllipSys3D employs the time independent mesh deformation matrix $\bar{\mathbf{U}} = \mathbf{F} \cdot \mathbf{A}$. This finding concludes the demonstration of this section.

Remark:
The mesh motion related to Equation 5.8 and the structural deformation vector **uh** is not included in the present considerations of defining a time independent mesh deformation matrix. This part of the mesh motion is neglected here because the transformation related to Equation 5.8 only conducts a rigid body motion where all mesh points are collectively rotated around the hub centre and then translated. The grid cells are not deformed and maintain their initial geometry.

5.4 The Implemented Coupling Schemes

5.4.1 Loose GSS Coupling without Force Scaling

Based on the guidelines given in Section 3.2 the loose coupling scheme of Figure 5.11 is designed with the aim of maintaining the second order time accuracy of the participating stand-alone solvers and restraining the error in energy transfer to a possibly third order accurate effect.

Therefore the time iteration starts with a second order accurate prediction of the structural deflections in HAWC2 using the formula of Equation 3.6 and the parameters $\alpha_0 = 1$ and $\alpha_1 = 1/2$.
After the deflections are transferred to EllipSys3D the mesh is deformed according to Section 5.2.2 which for small deformation angles corresponds to a mesh deformation using a time independent mesh deformation matrix $\bar{\mathbf{U}}$ as demon-

strated in Section 5.3.3. In order to obtain a second order time accurate mesh motion the ALE fluxes of Equation 2.12 are calculated using the quantities given in Section 3.2.6. The incompressible Navier Stokes Equations are then solved according to the SIMPLE or PISO algorithm using i subiterations until convergence is reached. As a final step inside EllipSys3D the aerodynamic forces are read out as explained in Section 5.2.4.
The extracted forces are then transferred to HAWC2 using Equation 3.5 in order to minimize the inherent energy error of the considered loosely coupled scheme. After the forces are applied on the HAWC2 structure the equations of motion are time integrated with the second order accurate Newmark scheme and then solved in a subiterative process using j subiterations until a converged result for the new deflection state $\mathbf{u}^{n+1}$ is found.

With respect to the performance of the presented coupling scheme we have to consider that the force and deflection transfer is non-conservative and that the erroneous energy introduced at the FS Interface could decrease the expected time accuracy and stability. The coupling scheme of Figure 5.11 thus comprises two sources of energy inaccuracy. The first source is related to the non-conservative load and motion transfer discussed in Section 5.3.1 and caused by the different spatial discretizations of the connected solvers. The second source is related to the inherent energy inaccuracy of any loosely coupled scheme discussed in Section 3.2.1 and caused by the inequality of Equation 3.3.

5.4.2 Loose GSS Coupling with Force Scaling

The coupling scheme of Figure 5.12 is similar to the one introduced in the preceding section but it additionally employs the force scaling of Equation 5.15 in order to establish an energy conservative force transfer across the highly non-matching FS interface. However, apart from a conservative load transfer the scheme still suffers from the inherent energy error of a loose coupling.

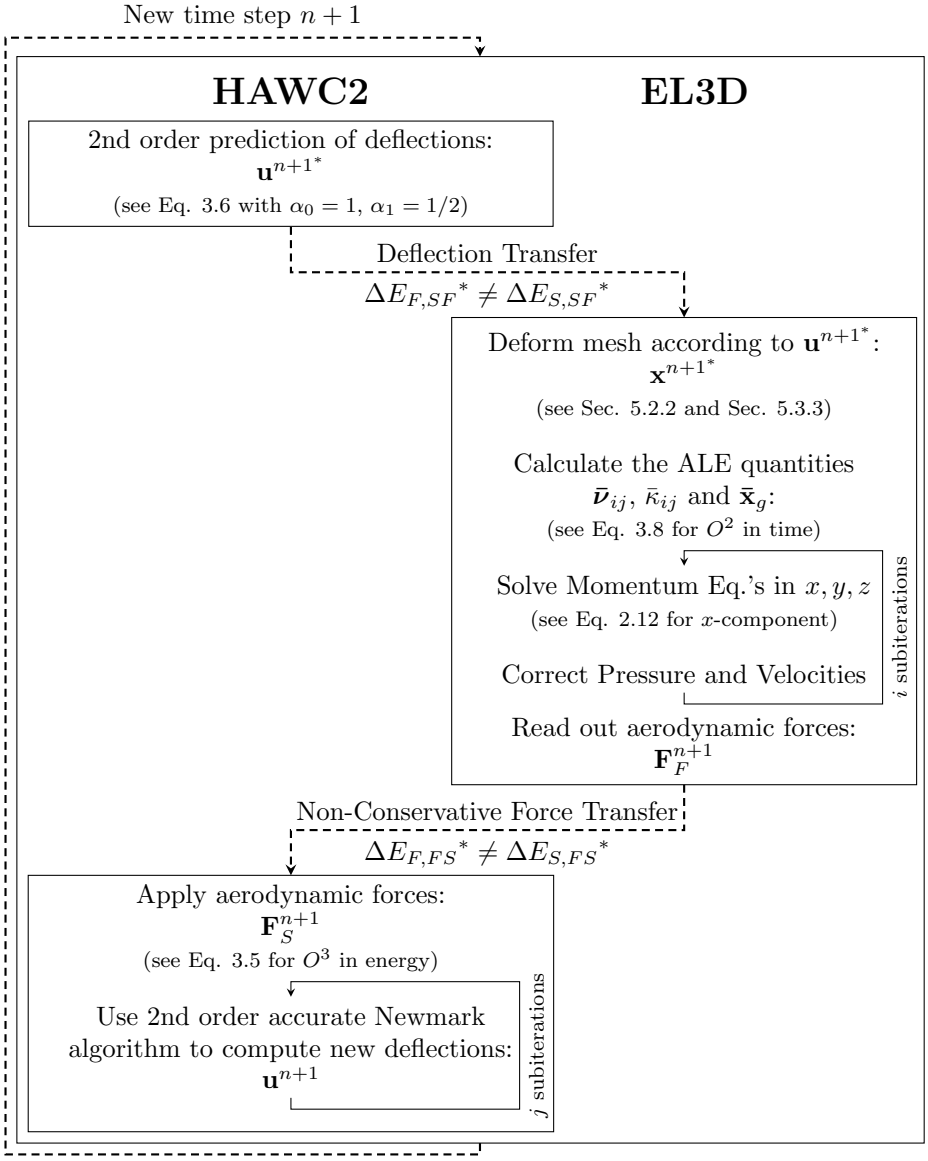


Figure 5.11: Implemented GSS Scheme without Force Scaling

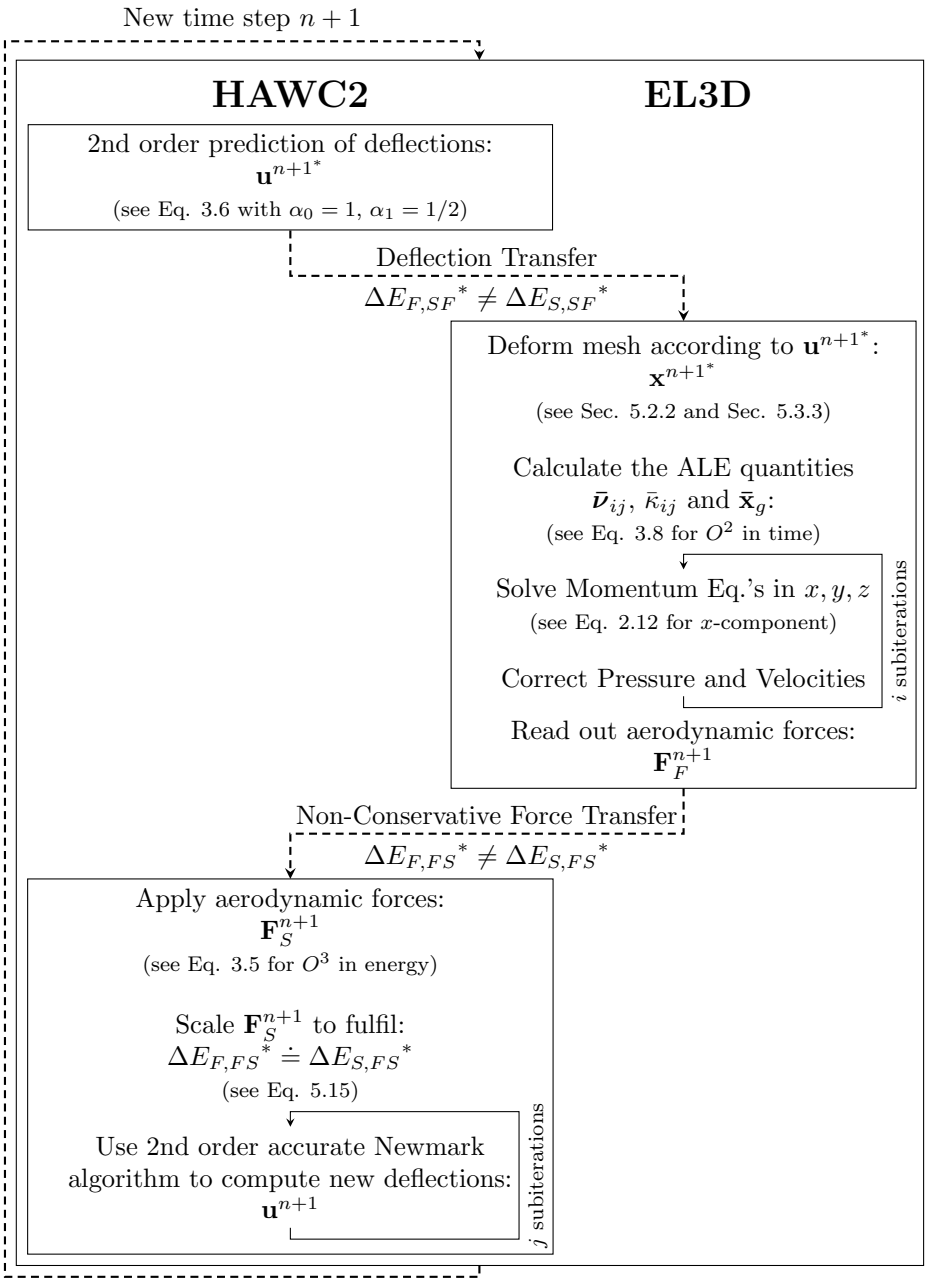


Figure 5.12: Implemented GSS Scheme with Force Scaling

5.4.3 Strong Coupling without Force Scaling

The inherent energy inaccuracy of the loose coupling schemes as discussed in Section 3.2.1 can only be circumvented by using a strong coupling scheme which employs a subcycling between the structural and fluid solver and thus reaches convergence between the actual deflection state and the respective aerodynamic forces. Figure 5.13 illustrates how the calculated deflections $\mathbf{u}^{n+1}$ of the Newmark time integration are fed back to the fluid solver in order to compute the corresponding aerodynamic forces $\mathbf{F}_F^{n+1}$ in a subiterative process. Those outer subiterations j continue until the convergence limit of the structural solver is met. Inside the fluid solver another subcycling is used in order to solve the Navier-Stokes Equation in correspondence to the SIMPLE or PISO algorithm respectively.

This strong coupling scheme removes the inherent energy inaccuracy of a loose coupling but still comprises the energy inaccuracy due to the non-conservative load and motion transfer discussed in Section 5.3.1.

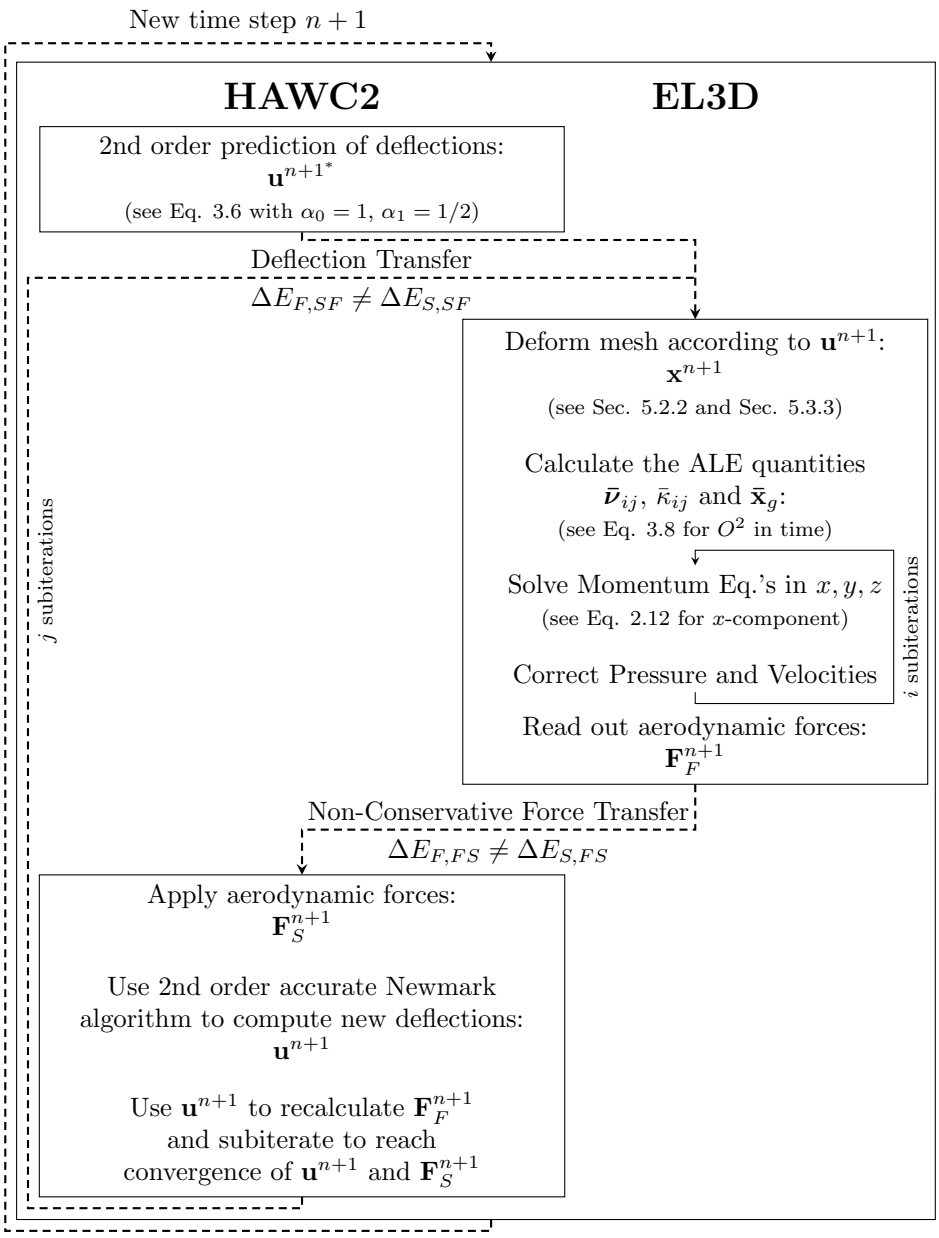


Figure 5.13: Implemented Strong Coupling Scheme without Force Scaling

5.4.4 Strong Coupling without Force Scaling and with Aitken Acceleration

The big drawback of employing a strong coupling scheme is the increase in computational costs. A strong coupling scheme demands several additional calls of the fluid solver and several additional solutions of the Navier-Stokes Equations which by far constitutes the computationally most expensive part of the present FSI coupling.
In literature several possibilities can be found in order to reduce the amount of subiterations j and to thus accelerate the convergence between structural and fluid solver. In Wall [10] the Aitken acceleration is presented which, due to its simplicity and efficiency, is well-known and widely used in strongly coupled FSI computations. In Figure 5.14 it can be seen that in the Aitken accelerated coupling scheme the computed structural deflections of the new time step $\mathbf{u}^{n+1}$ are not directly returned to the fluid solver but are relaxed by using a certain Aitken factor μ_j^{n+1}. An appropriate choice of this relaxation factor should reduce the amount of the subiteration j needed to reach convergence.

The Aitken factor is computed with the following equation

$$\mu_j^{n+1} = \mu_{j-1}^{n+1} + (\mu_{j-1}^{n+1} - 1) \cdot \frac{(\Delta\mathbf{u}_j^{n+1} - \Delta\mathbf{u}_{j+1}^{n+1})^T \Delta\mathbf{u}_{j+1}^{n+1}}{(\Delta\mathbf{u}_j^{n+1} - \Delta\mathbf{u}_{j+1}^{n+1})^2} \tag{5.27}$$

where $\mu_0^{n+1} = 0$ and where $\Delta\mathbf{u}_{j+1}^{n+1}$ is defined as the difference between the structural deflections of the current and the previous subiteration

$$\Delta\mathbf{u}_j^{n+1} = \mathbf{u}_j^{n+1} - \tilde{\mathbf{u}}_{j+1}^{n+1} \tag{5.28}$$

where $\tilde{\mathbf{u}}_{j+1}^{n+1}$ denotes the original deflection before the relaxation. The final relaxed deflection state $\mathbf{u}^{n+1}$ transferred to the fluid solver at subiteration j is then computed via

$$\mathbf{u}_{j+1}^{n+1} = (1 - \mu_j^{n+1}) \cdot \tilde{\mathbf{u}}_{j+1}^{n+1} + \mu_j^{n+1} \cdot \mathbf{u}_j^{n+1} \tag{5.29}$$

Remark:
It should be mentioned here that the condition of $\mu_0^{n+1} = \mu_{j,max}^n$ given in Wall [10] did not lead to satisfying results and the simplified condition of $\mu_0^{n+1} = 0$ was used instead. Since usually $\mu_{j,max}^n$ has a value close to one the preceding subiterative step starts with a relaxed deflection of $\mathbf{u}_{j+1}^{n+1} \approx \mathbf{u}_j^{n+1}$ which obviously

results in a very slow convergence process.

5.4.5 Strong Coupling with Force Scaling and with Aitken Acceleration

The strong coupling schemes of the previous sections have been implemented in order to circumvent the inherent energy inaccuracy of a loose coupling. As a final step it was now intended to incorporate the force scaling of Equation 5.17 into the strong coupling scheme and to thus also establish an energy conservative load and motion transfer. In this way both sources of energy inaccuracy could be omitted and the resulting coupling scheme could be considered as fully energy conservative.

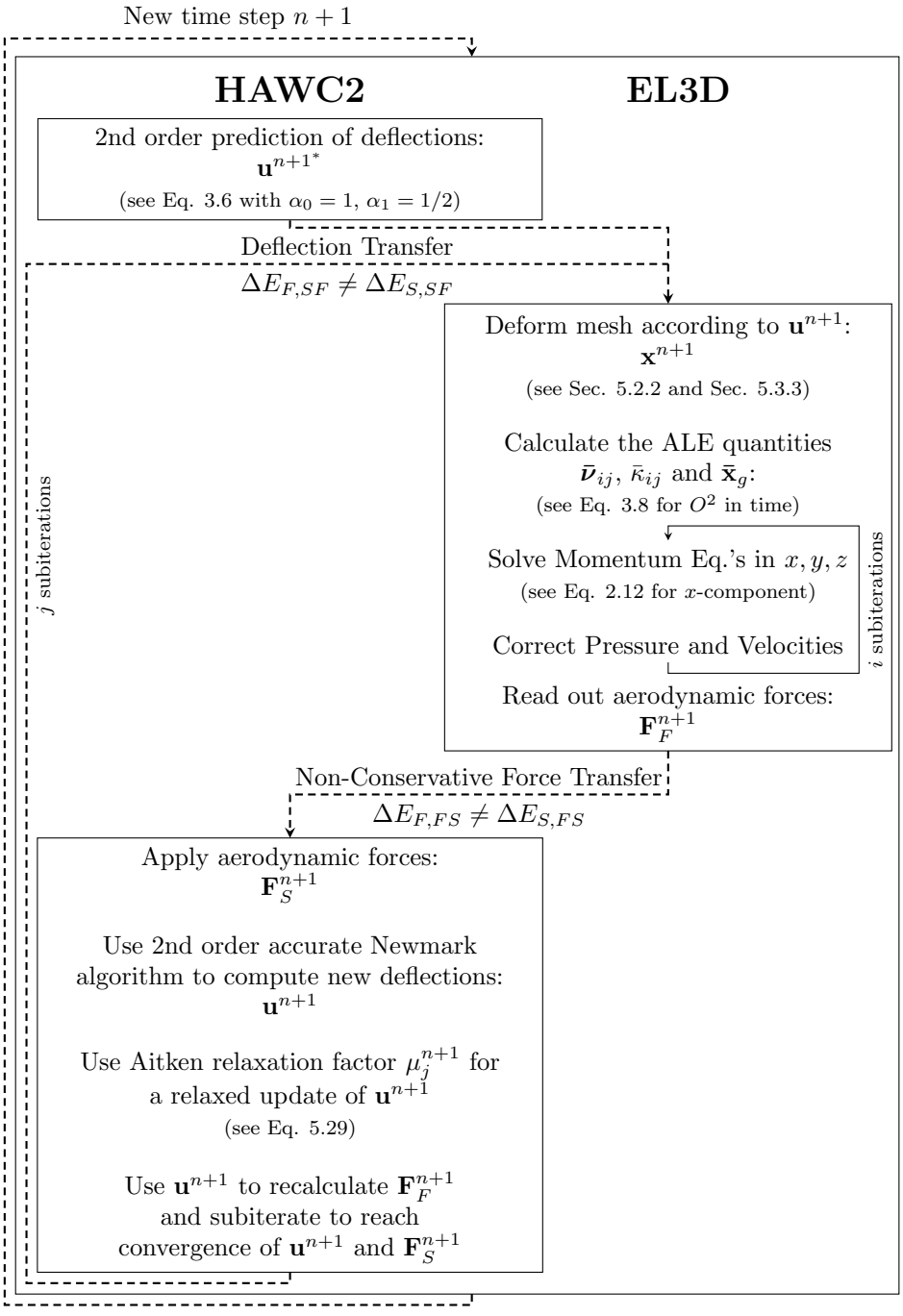


Figure 5.14: Implemented Strong Coupling Scheme without Force Scaling and with Aitken Acceleration

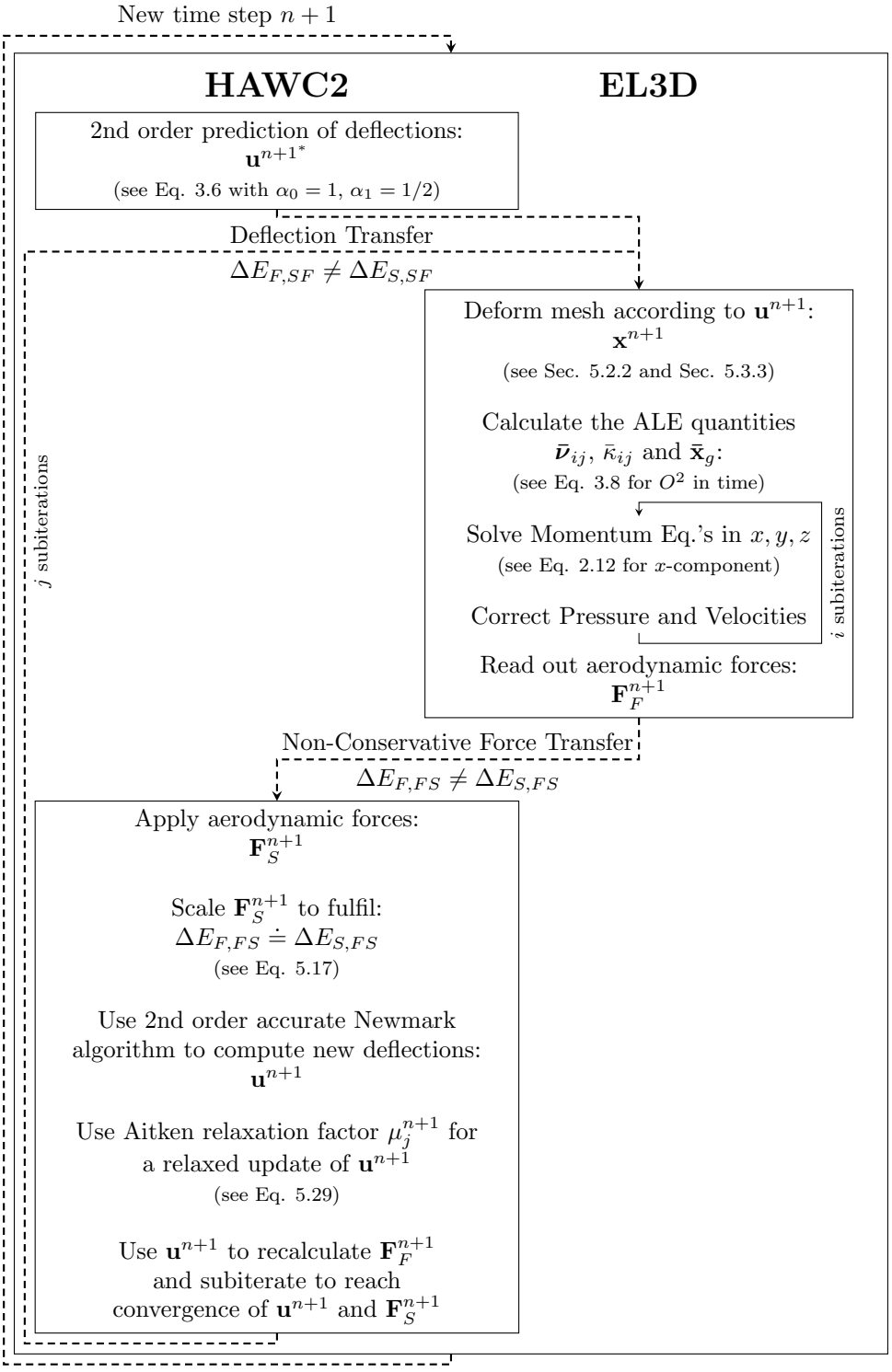


Figure 5.15: Implemented Strong Coupling Scheme with Force Scaling and with Aitken Acceleration

Chapter 6

Coupling of Other Models

Apart from the coupling of HAWC2 with EllipSys3D the developed coupling framework of Chapter 4 is also used to couple a simple 3-DOF structural solver with both the two-dimensional and the three-dimensional version of the fluid solver EllipSys. Skrzypiński employed the respective FSI couplings in [68] and [69] in order to investigate both stall-induced and vortex-induced vibrations of wind turbine blades during standstill. The additional coupling with a basic PI flap controller could be used to investigate the load reduction potential of an elastically mounted aerofoil section equipped with trailing edge flaps (such as done in Heinz [55]). Additional information about all participating models is given in Chapter 2.

The present Chapter is used to shortly discuss the implemented coupling related functions necessary to establish the desired FSI couplings between the respective models and to explicitly illustrate the elementary coupling scheme utilized to incorporate the 3-DOF structural solver into the aero-elastic computations.

6.1 Coupling Related Functions

In both fluid solvers EllipSys2D and EllipSys3D specific coupling related functions need to be implemented in order to facilitate a mesh motion in accordance to the given deflections of the 3-DOF structural solver. The translational and the rotational motion of the investigated aerofoil section as illustrated in Figure 2.3 is translated to the fluid mesh by moving all mesh points collectively in terms of a rigid body motion where all fluid cells maintain their original shape. In case the basic PI flap controller is connected to investigate the effects of an adaptive

trailing edge flap the mesh motion has to be able to represent the desired flap motion as well. In such an aero-servo-elastic computation the mesh points that describe the actual flap angle are determined by a linear interpolation between two CFD meshes which represent the minimum and the maximum flap angle deflection respectively.
In case the 3-DOF structural solver is connected to EllipSys the three integrated force components lift, drag and pitching moment need to be provided and applied to the structure of Figure 2.3. Therefore a routine is called in EllipSys which integrates the respective pressure and friction forces on the surface mesh of the entire aerofoil section. This integral is a line integral in the two-dimensional case and a surface integral in the three-dimensional case.
In case the basic PI flap controllers are connected to EllipSys some additional information has to be read out from the computed fluid mesh. For the flap controller that controls the flap in accordance to a Pitot tube measurement in front of the leading edge a certain coupling related function needs to first locate the position of the Pitot tube in the actual fluid mesh and needs to then continuously read out the local velocity components. For the flap controller that controls the flap in accordance to the pressure tap measurements on the aerofoil surface a certain coupling related function needs to first locate the positions of the respective pressure taps and needs to then continuously read out the local pressure values.

Considering the other models of Section 2.3 it should be mentioned that both the 3-DOF structural model and the basic PI flap controllers are especially developed for the present coupling. Since these codes do not exist as an independent stand-alone version there is no need to particularly determine the coupling related functions as defined in Section 4.5.

6.2 Elementary Coupling Scheme for 3-DOF Structural Solver

The 3-DOF structural solver presented in Section 2.3.1 employs a fourth order accurate but explicit Runge-Kutta-Nyström time integration scheme meaning that the deflections of the new time step are computed with the forces of the previous time step. This explicit treatment of the time integration leads to the usage of the elementary loosely coupled scheme described on Page 27. In the particular case of the coupling between the 3-DOF structural solver and the fluid solver

EllipSys2D or EllipSys3D this elementary loose coupling is implemented as illustrated in Figure 6.1.

It might be interesting to note that in contrast to the coupling between HAWC2 and EllipSys3D both the force and the deflection transfer of the presented coupling can be considered as fully energy conservative. The aerodynamic forces transferred to the structural solver are the integrated forces over the entire aerofoil section and the three resulting components of lift, drag and moment are thus exactly determined and can be directly applied on the single structural node of the 3-DOF model. The energy released by the fluid is exactly the same as the energy absorbed by the structure. The same can be said in the opposite direction when the three structural deflections are transferred to the fluid solver by a rigid body motion of the entire aerofoil section.

6.3 Results

It was not in the scope of the present work to employ the FSI couplings of the present Chapter for further investigations. However, as mentioned before, respective applications and results can be found in the publications of Skrzypiński and Heinz [68], [69], [55].

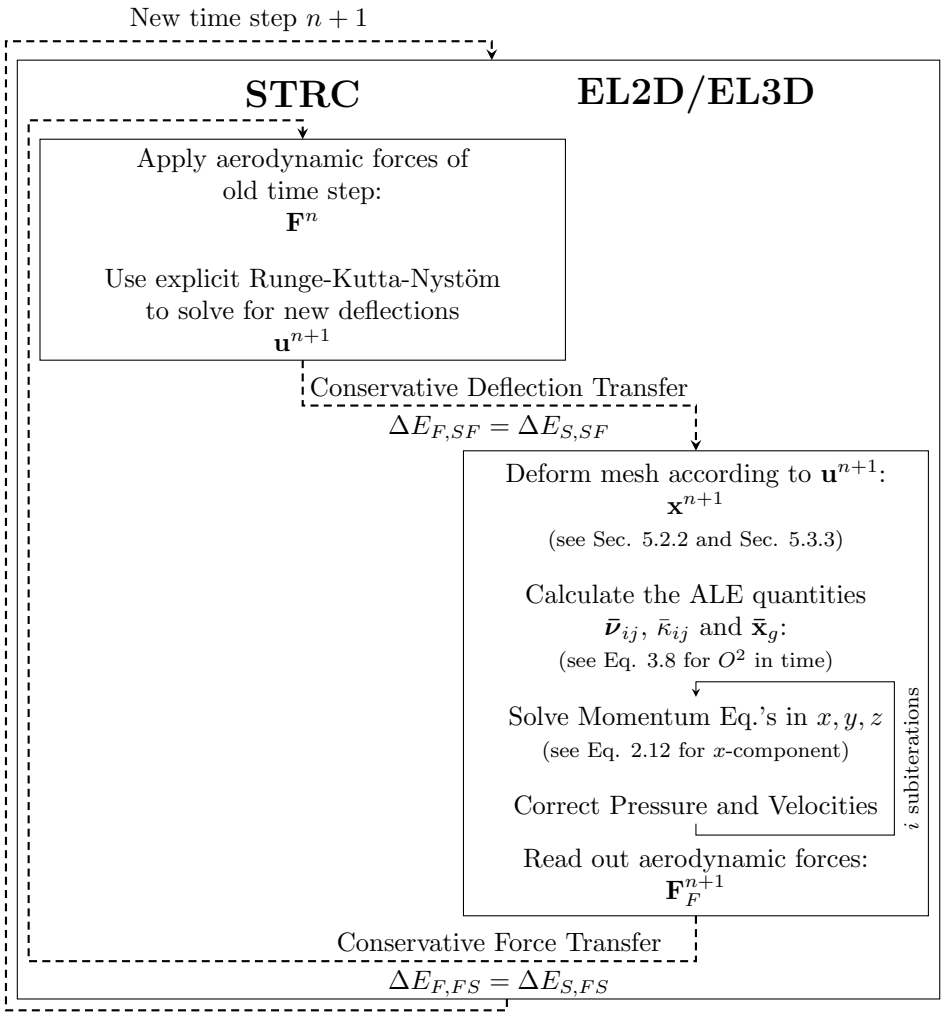


Figure 6.1: Implemented Elementary Coupling Scheme for the 3-DOF Structural Solver

Chapter 7

Full Rotor Simulations

7.1 Investigated Wind Turbine Model

The wind turbine model used for the full rotor simulations of the present chapter is the NREL 5MW reference wind turbine as documented in Jonkman [58]. The model was developed with a focus on offshore applications characterized by the relatively low hub height chosen to minimize the overturning moment acting on an eventual offshore substructure. The objective of designing this reference turbine was to provide a baseline wind turbine model with a freely and fully accessible data set in order to allow any interested researcher in the world to model this wind turbine and to thus contribute to the offshore research with easily comparable results. In the present work the NREL 5MW reference wind turbine was selected in order to obtain public available results of common interest from a well documented wind turbine model.

7.1.1 Structural Model

The NREL 5MW reference turbine is a three bladed upwind turbine with a hub height of 90 m, a hub diameter of 3 m and a blade length of 61.5 m. Assuming a cone angle of 0 ° this results in a rotor diameter of 126 m and a minimum air gap of 17 m between blade tip and ground. The rotor overhang is chosen to be 5 m and a cone angle of 2.5 ° is proposed to obtain a sufficient tower clearance. The given mass distribution of the turbine blades sums up to a total weight of nearly 18.000 kg per blade, together with an estimated hub weight of nearly 57.000 kg the total weight of the rotor adds up to 110.000 kg. Further information about the exact structural properties of the blades and the other structural components like the turbine tower can be found in Jonkman [58].

The structural model in HAWC2 is built up as illustrated in Figure 2.1. In order to appropriately describe the blade deflections of the NREL 5MW reference turbine the model employs nine bodies, eighteen beam elements and thirty aerodynamic sections for each wind turbine blade. The tower and the shaft are modelled using only one single body respectively.

7.1.2 Aerodynamic Model

The aerodynamic properties of the NREL 5MW reference turbine are based on the aerodynamics used in the DOWEC study of Kooijman [75]. Starting with a cylindrical shape at the root section the aerofoil shape gradually merges into a DU40 aerofoil with a relative thickness of 40 %. From the DU40 aerofoil at the radial blade position of approximately $r = 10\,\mathrm{m}$ the relative thickness is gradually reduced until the DU21 aerofoil with a relative thickness of 21 % is used at the radial blade position of approximately $r = 40\,\mathrm{m}$. At the remaining outer part of the blade the NACA64-618 aerofoil with a relative thickness of 18 % is used.

For the traditional BEM based simulation tools the necessary lift, drag and moment coefficients of the respective aerofoils can be found in the appendix of Kooijman [75]. In Jonkman [58] those two-dimensional aerofoil coefficients are then corrected in order to account for certain three-dimensional effects. Therefore, the lift and drag coefficients are corrected for rotational stall delay using the Selig and Eggars method and the drag coefficients are further corrected using the Viterna method while assuming an aspect ratio of 17 %.

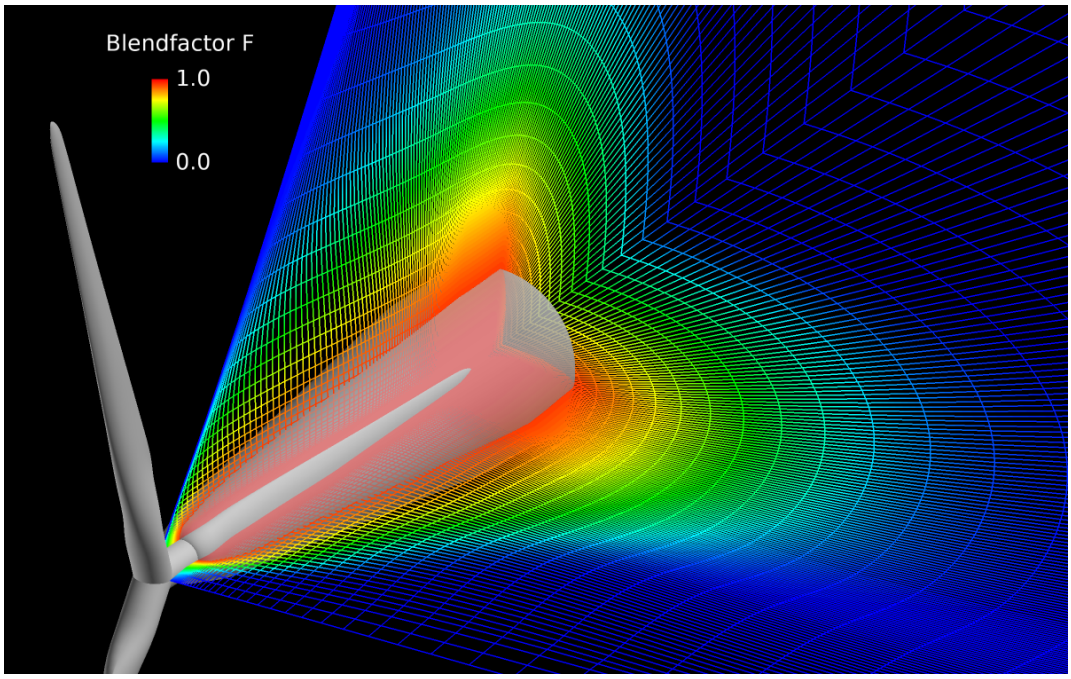
Using the high-fidelity FSI coupling between HAWC2 and EllipSys3D for the aero-elastic computations the aerodynamic modelling is no longer depending on the above mentioned aerofoil data corrections which intent to include three-dimensional effects on originally two-dimensional aerofoil coefficients. Instead, the aerodynamic model of EllipSys3D is solely depending on the given aerofoil and blade geometry. No further aerofoil data is needed.

After the surface geometry of the NREL 5MW turbine is defined an appropriate fluid mesh has to be generated around this surface. The chosen dimensions of the fluid mesh utilized for the FSI simulations of the present chapter are based on the experience of previous aerofoil and rotor computations [32]. The generated surface mesh is using 256 mesh cells to enclose the blade in the chordwise

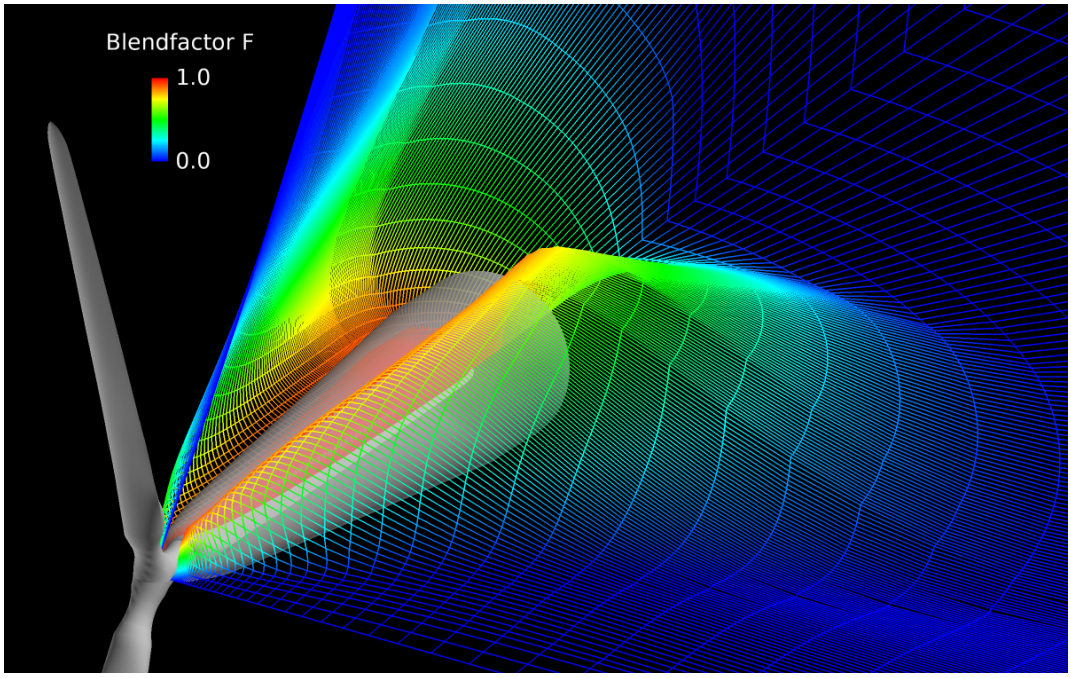
direction perpendicular to the blade axis and 128 mesh cells to cover the blade in the spanwise direction along the blade axis. In the normal direction to the blade surface 128 mesh cells are used to march the grid to the farfield, starting with a cell size of 10^{-6} m in order to assure a value of $y^+ < 2$. This results in a fluid mesh with a total amount of around 14 million grid cells where the rotor is located in the centre of a spherical domain with a total diameter of approximately twenty times the rotor diameter.
In Figure 7.1a the distribution of fluid cells in the spanwise and in the normal direction are shown for a certain blade. In Figure 7.2a the distribution of fluid cells in the chordwise direction perpendicular to the blade axis are shown.

However, the respective figures do not only illustrate the chosen cell distribution along a certain turbine blade b, they also indicate the chosen blendfactors $\mathbf{F}^b$ utilized to accomplish the mesh deformations described in Section 5.2.2. Figure 7.1a shows the blendfactors in the undeformed mesh configuration. At a certain area around the blade the blendfactors hold the value $F = 1$. As seen from Equation 5.7 this is the area where the mesh points will entirely follow the structural deformation of the blade. In Figure 7.1a the area of this rigid body motion is additionally indicated with a semitransparent white cone. Towards the farfield of the mesh the blendfactors are then gradually reduced until they reach the minimum value of $F = 0$ where the mesh points maintain their initial position and do not feel the blade motion at all. The transition area with $0 < F < 1$ has to be wide enough to facilitate a smooth transition from the fully deformed to the non-deformed state, however, at the same time it has to be narrow enough to not interfere with the mesh deformation of the neighbouring blades.

For the aero-elastic computations of the present chapter it was desired to simulate an emergency shut-down where the wind turbine blades have to turn to a final pitch angle of $\theta = 90\,^\circ$. This extraordinary and extreme pitch setting constitutes an ultimate challenge for the implemented mesh deformation algorithm. In order to demonstrate that even this extreme pitch setting can be mapped satisfactorily some additional figures are shown in order to illustrate the respective deformed mesh configuration in different perspectives. Figure 7.1b nicely illustrates how the mesh points close to the blade surface entirely follow the blade motion while the mesh points with blendfactor $F = 0$ maintain their original position. Figure 7.2a and Figure 7.2b show the deformed mesh cells at the thickest part of the blade where due to the relatively narrow transition region and due to the rel-

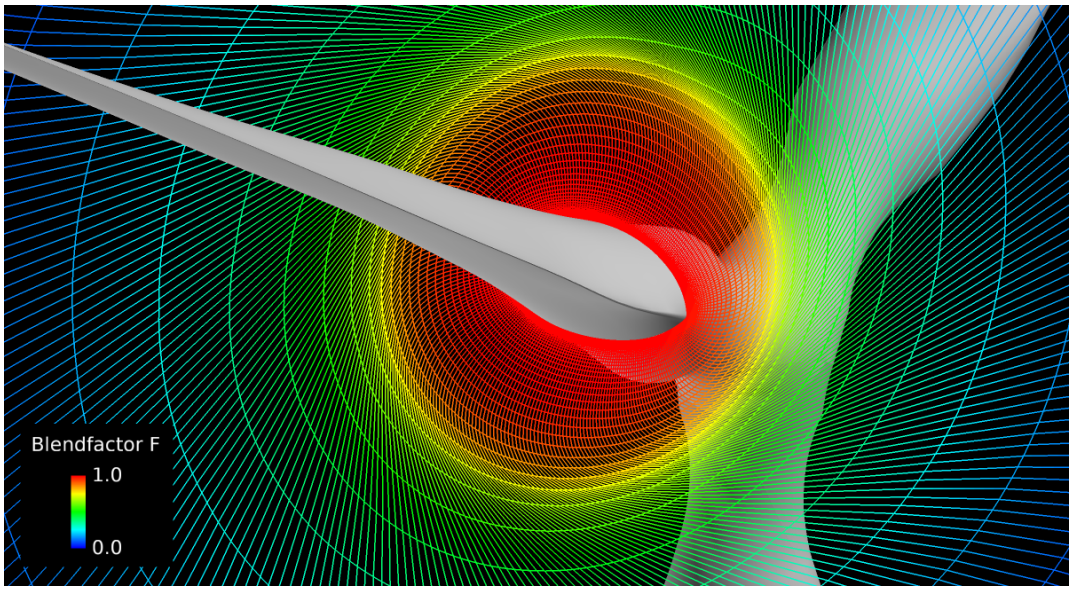


(a) Fluid Mesh and Blendfactors at a Pitch Angle of $\Theta = 0°$

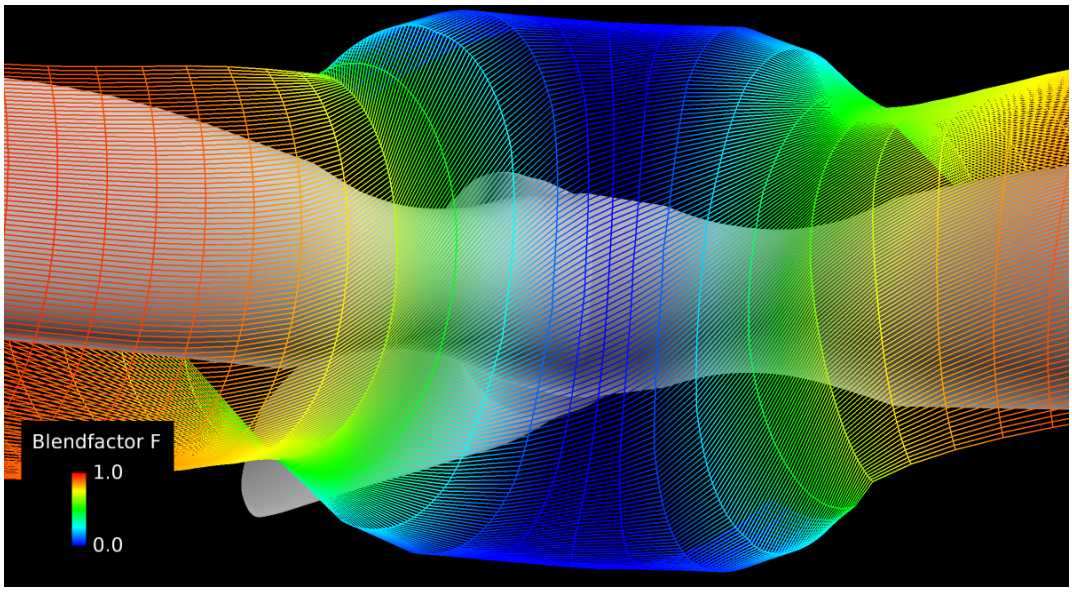


(b) Fluid Mesh and Blendfactors at a Pitch Angle of $\Theta = 90°$

Figure 7.1: Fluid Mesh and Blendfactors for the NREL 5MW Reference Turbine



(a) Fluid Cells Orthogonal to Blade Surface



(b) Fluid Cells Parallel to Blade Surface

Figure 7.2: Fluid Mesh and Blendfactors for the NREL 5MW Reference Turbine with $\Theta = 90°$

atively large movements of the grid points the demands on the deformed mesh cells are highest. However, although the requirements for the deformed mesh are very high in this area it can be seen that in both the orthogonal and the parallel plane the fluid cells still reveal a satisfactory geometry with a moderate skewness only.

It should be finally mentioned here that the aerodynamic model of the present FSI coupling between HAWC2 and EllipSys3D does not account for the aerodynamic influences of tower and nacelle. As seen in the previously discussed figures the aerodynamic model only accounts for the rotor of the NREL 5MW turbine including the turbine blades and the hub. While the tower and the nacelle are considered in the structural model and are there contributing to the structural response of the FSI computation they are not modelled in the aerodynamic part and the aerodynamic influence of those components is thus neglected. However, for the present investigations the aerodynamic influence of the nacelle and the tower is considered to be rather small.

The computations of the present chapter are conducted under the assumption of a fully turbulent flow. The inflow is chosen to be homogeneous over the entire rotor meaning that the effect of wind shear is not considered.

7.1.3 Variable Speed, Variable Pitch Controller

Most of the full rotor computations of the present chapter are not only employing the coupling between a structural solver and a fluid solver, they also employ a control model that assures that the wind turbine is kept in a realistic operational state during the entire simulation. The presented FSI computations are thus not only simulating the aero-elastic response but in particular the aero-servo-elastic response of the wind turbine rotor.

The utilized controller is the suggested variable speed, variable pitch controller for the NREL 5MW reference turbine as documented in [58]. Below rated speed the controller acts on the generator torque in order to optimize the power capture by maintaining the optimal tip speed ratio. Above rated speed a collective pitch-to-feather mechanism leaves the optimal operational point in order to limit the power output and the generator speed. In order to keep the power output in the region above rated speed constant the generator torque is set inversely

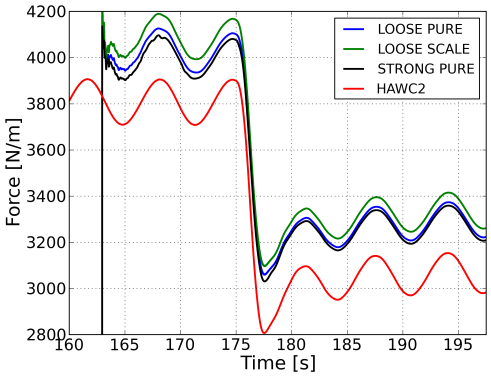
proportional to the generator speed. In both control regions, above rated and below rated speed, the sole input to the controller is the actual generator speed.

Since the employed control model is not using any continuous aerodynamic input and does not need to interact directly with the fluid solver it was not necessary to connect the control model via the developed Python coupling framework of Chapter 4. Instead it was decided to connect the control model via the already existing internal coupling interface of HAWC2. In this way a well-tested implementation of the controller could be used since this controller was already extensively employed in the traditional HAWC2 simulations of the stand-alone solver.

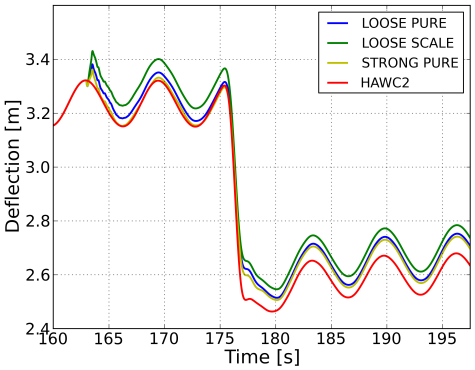
7.2 Assessment of the Implemented Coupling Schemes

In Chapter 3 the advantages and disadvantages of loose and strong coupling schemes were discussed. It was mentioned that the computationally cheap loosely coupled algorithms induce an inherent energy inaccuracy which is frequently accused of triggering numerical instabilities. The explicit character of a loose coupling, where the aerodynamic loads are only calculated on the predicted deflection state, was blamed for reducing the time accuracy of the overall FSI computation to the order of one. Strong coupling schemes do not show those characteristics, however, the additional subiterations increase the computational costs decisively and several authors within the field of aeroelasticity argue that it is indeed possible to establish a sufficiently stable and higher order accurate loosely coupled algorithm. In the present coupling between HAWC2 and EllipSys3D several coupling schemes were thus implemented in order to investigate the influences of those recent findings in literature. The respective coupling schemes of Section 5.4 also include a variation in the implemented force transfer incorporating both a non-conservative force transfer and a conservative force transfer achieved by a simple force scaling.

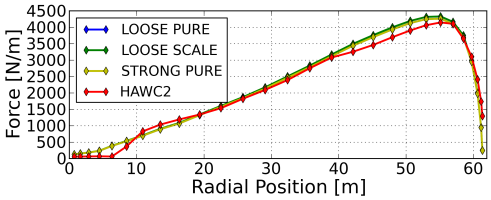
This section tries to assess the implemented coupling schemes during a typical aero-elastic simulation of an entire wind turbine rotor. Within the first three subsections it will be shown that the loosely coupled GSS scheme of Section 5.4.1 without force scaling seems to be the best choice for the desired FSI simulations.



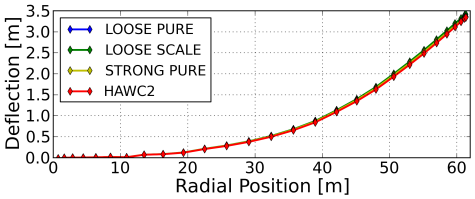
(a) Normal Force in $BSCS_S$ at $r = 75\,\%$



(b) Flapwise Blade Deflection in $BSCS_S$ at $r = 95\,\%$



(c) Normal Force in $BSCS_S$ at $t = 168\,s$



(d) Flapwise Blade Deflection in $BSCS_S$ at $t = 168\,s$

Figure 7.3: Comparison of Different FSI Coupling Schemes with Stand-Alone HAWC2 Computations, NREL 5MW Turbine, $U_\infty = 8\,m/s$, Uniform Inflow, Pitch Step from $\Theta = 0\,^\circ$ to $\Theta = 2\,^\circ$

In Section 7.2.4 a time accuracy investigation will then emphasize that the selected loose coupling scheme is indeed capable of maintaining the 2nd order time accuracy of the stand-alone solvers.

7.2.1 Comparison with Traditional HAWC2 Computations

In a first test case the aero-elastic response of the NREL 5MW reference turbine was simulated during a step change in pitch angle from $\Theta = 0\,^\circ$ to $\Theta = 2\,^\circ$. In this basic set-up the turbine was subjected to a uniform and symmetric windfield of $U_\infty = 8\,\mathrm{m/s}$ with laminar inflow, with no wind shear and with a yaw and tilt angle of $0\,^\circ$. The angular velocity of the turbine was set to the constant value of $\omega = 0.968\,\mathrm{rad/s}$. In terms of simplicity the structural flexibility was limited to the wind turbine blades only, other components like tower and rotor shaft were assumed to be stiff. The FSI simulations are carried out with a time step size of $\Delta t = 0.0025\,\mathrm{s}$.

In Figure 7.3a the normal force in $BSCS_S$ coordinates at a radial blade position

of $r = 75\,\%$ and in Figure 7.3b the flapwise blade deflection in $BLCS_S$ coordinates at a radial blade position of $r = 95\,\%$ are plotted over time. The results of the traditional HAWC2 computation using the BEM based aerodynamic model are shown together with the results of the new high-fidelity FSI coupling between HAWC2 and EllipSys3D using the three coupling schemes of Section 5.4.1, Section 5.4.2 and Section 5.4.3. The coupling schemes without force scaling are denoted as the *pure* versions. The FSI simulations are carried out using the start-up procedure explained in Appendix B.1 where all simulations are first initiated with the computationally cheap BEM model and where the computationally heavy CFD computations are only used after the initial transients are dampened out. In the given example the aerodynamic models are switched at the time instant $t = 163\,s$.

A first rough comparison between the two different loose coupled versions indicates that both versions predict the dynamics of the aero-elastic response in a very similar way. The force scaling and thus the enforced energy conservative force transfer does not seem to have a major influence on the results. Since the energy level of the fluid solver turned out to be slightly higher than the energy level of the structural model the scaling of the forces as proposed in Equation 5.15 results in a slight increase of the aerodynamic forces applied on the structure. More details about the different energy levels and a final assessment of whether the force scaling is recommendable or even necessary will be given in Section 7.2.3.

The results of the strong coupling scheme without force scaling are now compared to the results of the loose coupling scheme without force scaling. The comparison shows that both the dynamic behaviour in general as well as the magnitude of the forces and deflections agree very well. However, the established equilibrium between forces and deflections is slightly below the level of the respective loosely coupled simulation. The strong coupling does not seem to have any decisive effect on the dynamics and the stability of the current test case. Instead, the dynamics seem to be highly determined by the motion of the relatively heavy turbine structure. In the terms of Section 3.3 it can be stated that the density ratio ρ_F/ρ_S of the considered aero-elastic simulation seems to be sufficiently low.

Finally the results of the three investigated coupling schemes are compared to the results of the traditional HAWC2 simulation using BEM theory. The first impression while observing Figure 7.3a could be that the new FSI simulation tool gives a rather distinct difference in the force level. However, observing the force distribution over the entire blade demonstrates that this difference is not based on

an alleged inaccuracy of the coupling methods. Figure 7.3c shows the force distribution over the entire blade at a certain time instant $t = 168\,\mathrm{s}$ where it can be seen that the normal forces of the traditional HAWC2 computations are slightly reduced at the outer blade section with $r > 70\,\%$. The same observations can be made at any other time instant $t \neq 168\,\mathrm{s}$ and the differences are also existent in the more detailed aero-elastic computations of Section 7.3.2. The consistent observations demonstrate that the reasons for the discrepancies can be found in the different aerodynamic modelling of the wind turbine blades. While the CFD computations of the tested coupling schemes are based on the three-dimensional geometry of the rotor blades the BEM computations of the traditional HAWC2 simulation are based on two-dimensional aerofoil data. The direct comparisons between the models now show that for $r > 70\,\%$ the provided aerofoil data of the NREL 5MW reference turbine does not exactly resemble the aerodynamics of the respective CFD computations.
Due to the slightly reduced normal forces in the traditional HAWC2 simulations the respective blade deflection of Figure 7.3d are also slightly reduced towards the tip. A more detailed discussion about the different force levels between CFD and BEM computations is given in Section 7.3.2.

At the end of this section it should be emphasized how well the results of the different models agree. The slight differences in the aerodynamic forces at the outer blade section can be easily reduced by choosing slightly different aerofoil data for the BEM based HAWC2 computations. Apart from this difference the general behaviour of the models agree very well. Considering the fact that the underlying aerodynamic formulations of HAWC2 are completely different this agreement is particularly remarkable.

Remark:
Unfortunately, the strong coupling scheme of Section 5.4.5 with integrated force scaling could not be tested successfully. Within the subiterative process of the strong coupling the force scaling of Equation 5.17 showed a very strong influence on the energies $\Delta E_{F,FS}$ and $\Delta E_{S,FS}$ at the FS interface. As a consequence the scaling ratio $\Delta E_{F,FS}/\Delta \tilde{E}_{S,FS}$ continuously varied between values above and below one, and the constant change between up-scaling and down-scaling resulted in heavily oscillating force signals. Several attempts to smoothen the oscillations did not lead to a satisfying result.

7.2.2 Computational Costs of the Strong Coupling Schemes

A strong coupling scheme circumvents the introduction of the inherent erroneous energy of a loosely coupled algorithm. As explained in Section 3.2.1 this erroneous energy stems from the fact that the aerodynamic forces of a loosely coupled algorithm are exclusively based on the predicted deformation state. Within the subiterations of a strong coupling the aerodynamic forces are successively recomputed on the actual deflection state and the erroneous energy of Equation 3.3 becomes equal to zero. However, this benefit comes with a major increase of computational costs.

Several tests have been carried out in order to minimize the computational costs of the strongly coupled simulations. While the Aitken acceleration of Section 5.4.4 showed only little influence on the overall simulation time it was found that, referring to the flow chart of Figure 5.14, the chosen number of fluid subiterations i has a decisive impact on the time consumption of the FSI computations and that the maximum number of fluid subiterations i_{max} should be defined as a function of the actual outer subiterations j.
In order to demonstrate these findings the results of three different acceleration strategies are shown in Figure 7.4. The Figure illustrates the convergence behaviour of a strongly coupled simulation by showing the normal force signal at a blade position of $r = 75\,\%$ during the time interval of two full time steps $[t^n,t^{n+2}]$. However, instead of plotting the full time steps on the x-axis the total amount of required fluid subiterations i is shown. The distinct peaks in the force signals identify the first subiteration of a new time step, they are caused by the semi-implicit treatment of several terms in the fluid solver where the values of the first subiteration are still based on the values of the old time step.
In case the maximum number of fluid subiterations is defined as $i_{max}^{j=1,2,\cdots} = 1$ meaning that the aerodynamic forces are exchanged after each subiteration i of the fluid solver, a total amount of $i_{tot} = 33$ fluid subiterations is needed to simulate the two time steps. The distinct zig-zag pattern of the force signal shows that the coupled simulation has certain problems to reach convergence. A decisive improvement could be achieved if the maximum number of fluid subiterations is defined with $i_{max}^{j=1,2,3} = 2$ and $i_{max}^{j=4,5,\cdots} = 1$ which means that during the first three outer iterations j the aerodynamic forces are only exchanged after every second fluid subiteration. This modification could reduce the total amount of subiterations to $i_{tot} = 21$ and corresponds to a decrease of $35\,\%$. Several other choices

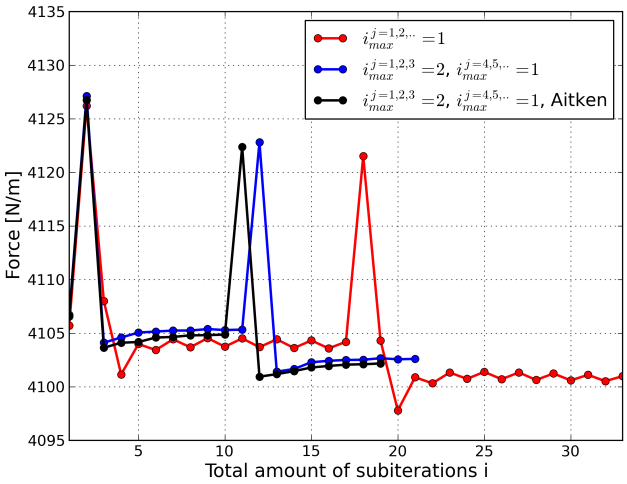


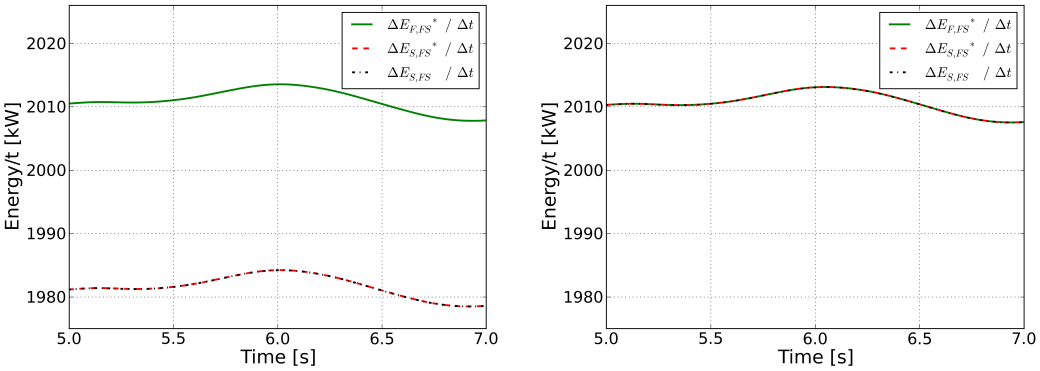
Figure 7.4: Force Convergence for Different Acceleration Strategies

and combinations for i_{max}^{j} have been examined as well, however, the presented sequence of $i_{max}^{j=1,2,3} = 2$ and $i_{max}^{j=4,5,\cdots} = 1$ led to the best results.
By keeping the same iteration strategy but additionally using the Aitken acceleration of Section 5.4.4 the total amount of subiterations could be reduced with another 10 %. The plotted values are extracted at a random time interval of the FSI simulation and represent a good picture of the general tendencies.

Since the solution of the Navier-Stokes equations within the fluid subiterations i represent by far the computationally most expensive part of the present FSI coupling between HAWC2 and EllipSys3D, the findings of Figure 7.4 can be directly related to the overall computational costs of the entire FSI simulation. Considering the fact that a conventional CFD computation in a loosely coupled FSI simulation employs around six subiterations per time step we can thus state that an optimized and Aitken accelerated strong coupling scheme still increases the computational costs by 50 % to 70 % percent.

7.2.3 Energy Investigation during Full Rotor Simulation

The discussion of the present section is based on the several energies defined and utilized in Section 3.2.1, Section 5.3.1 and Section 5.3.2. The investigations will help to understand and quantify the influences of the different energies and will thus help to choose the most appropriate coupling scheme from Section 5.4 for the subsequent simulation test cases. In the subsequent figures all presented energies are divided by the employed time step size of the FSI simulation in order to get a



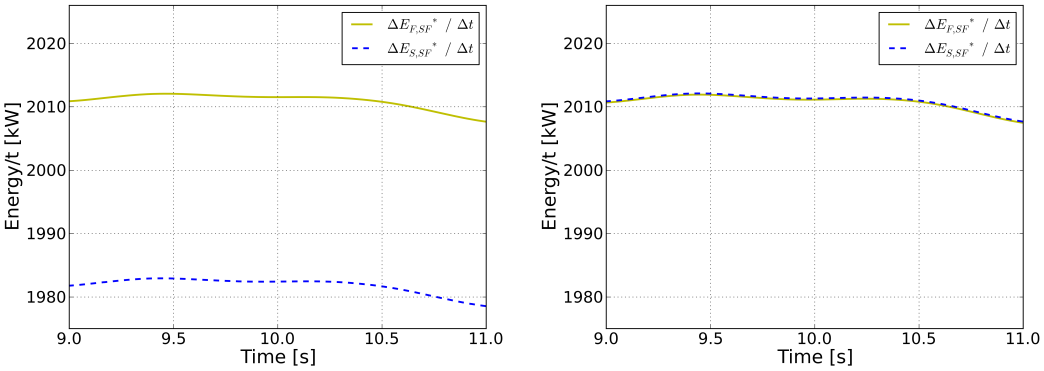
(a) Without Force Scaling

(b) With Force Scaling

Figure 7.5: Energy Levels during Force Transfer

better comparable and a better conceivable quantity. The energies are monitored during a full rotor FSI simulation using a uniform and symmetric windfield of $U_\infty = 8\,\mathrm{m/s}$, an angular velocity of $\omega = 0.968\,\mathrm{rad/s}$ and a fixed pitch angle of $\Theta = 0\,°$.

Figure 7.5a shows the quantities during a loosely coupled simulation without force scaling. The energies $\Delta E_{F,FS}{}^*$ and $\Delta E_{S,FS}{}^*$ are defined in Section 5.3.1 and represent the energies that are released by the fluid and received by the structure during the non-conservative load transfer. Both energies are based on the predicted deflection state and are thus indeed comparable. It can be seen that the energy released by the fluid is approximately 1.5% higher than the corresponding energy received by the structure. However, if this gap in energy can be assigned to the non-conservative load transfer will be discussed later.
Figure 7.5b shows the respective results for a loose coupling scheme with integrated force scaling. As expected, the scaling increases the applied aerodynamic forces on the structure and lifts the energy level $\Delta E_{S,FS}{}^*$ to the same height as $\Delta E_{F,FS}{}^*$.
The third quantity shown in the graphs of Figure 7.5 is the energy $\Delta E_{S,FS}$. This energy is based on the converged deflection state and represents the true energy received by the structure. Especially for an energy conservative load transfer with $\Delta E_{F,FS}{}^* \doteq \Delta E_{S,FS}{}^*$ the difference between $\Delta E_{S,FS}{}^*$ and $\Delta E_{S,FS}$ is directly comparable to the energy difference of Equation 3.4 and is thus providing a good indication of the inherent energy error of the loose coupling scheme. It is this energy error that could be omitted by using a strongly coupled algorithm. In both graphs of Figure 7.5 the gap between $\Delta E_{F,FS}{}^*$ and $\Delta E_{S,FS}$ seems to be



(a) Without Force Scaling

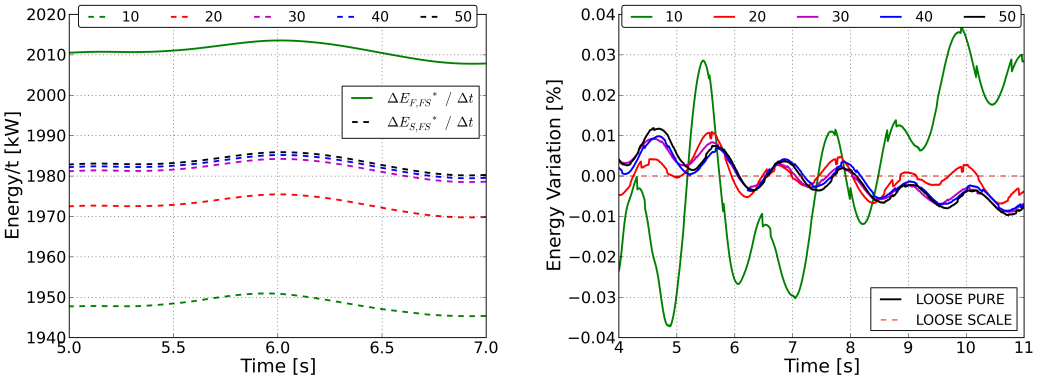
(b) With Force Scaling

Figure 7.6: Energy Levels during Deflection Transfer

rather small and negliable, a more detailed discussion about the significance of this energy error is given later in this section.

In Figure 7.6 the energies $\Delta E_{S,SF}{}^*$ and $\Delta E_{F,SF}{}^*$ are shown. The energies are defined in Section 5.3.2 and represent the energies that are released by the structure and received by the fluid when the structural deflections are transferred. The energies have been recorded in order to also understand the effect of the chosen deflection transfer on the coupling. It can be seen that the curves show the same tendencies as the curves of Figure 7.5. In the simulation without force scaling the energy gap between fluid and structure is approximately 1.5% and the implemented force scaling is then closing the gap between the respective energies. Thus, the force scaling seems to be a sufficient tool in order to run the coupled solvers on the same energy level. There is no need for a respective scaling of the transferred deflections.

As explained in Section 5.3.1 the force scaling was implemented in order to obtain an energy conservative force transfer via the highly non-matching meshes of fluid and structural solver. In Figure 7.5a it could be seen that for a simulation without force scaling the gap between the energy $\Delta E_{F,FS}{}^*$ released by the fluid and the energy $\Delta E_{S,FS}{}^*$ received by the structure was approximately 1.5%. If the observed energy gap is indeed due to the different spatial discretizations of the coupled solvers the energy gap should diminish with an increased amount of employed aerodynamic sections. A sensitivity analysis has been carried out and the results for a loose coupling scheme without force scaling are shown in Figure 7.7a. The energy $\Delta E_{F,FS}{}^*$ released by the fluid is not noticeably affected by the



(a) Energy Levels during Force Transfer

(b) Energy Variations

Figure 7.7: Energy Investigations Depending on Amount of Aerodynamic Sections per Blade

chosen amount of aerodynamic sections per turbine blade, but it can be seen that the energy $\Delta E_{S,FS}{}^*$ received by the structure successively rises for the amount of 10, 20, 30, 40 and 50 aerodynamic sections. However, the energy of the structure stabilizes at a certain level and retains the previously found energy gap of 1.5%. It can thus be concluded that the remaining difference in the energy levels is not related to the non-conservative force transfer and other reasons have to be found. The author therefore assumes that the remaining energy gap between fluid and structural solver is related to the very different modelling approach within the respective solvers. The wind turbine model of the fluid solver experiences aerodynamic forces on the entire rotor. Especially at the blade root and at the hub region the CFD model captures a certain loading which is definitely not existing in the turbine model of the structural solver. Those differences in the loading are thus assumed to cause the slightly differing energy levels and the observed energy gap of 1.5%.

Accepting the fact that the coupled solvers run on slightly different energy levels it is interesting to see if the higher energy level of the fluid solver somehow influences the stability of the coupled simulations by e.g. pushing erroneous energy from the fluid to the structure. In order to investigate this the energy $\Delta E_{F,FS}{}^*$ released by the fluid is subtracted from the energy $\Delta E_{S,FS}{}^*$ received by the structure. By subsequently subtracting the temporal mean of this difference and relating the result to the absolute value of the mean we obtain a relative quantity which indicates how much energy is pushed from the fluid to the structure and vice versa. This relative quantity or, in other words *energy variation*

is shown in Figure 7.7b where different simulation runs with a varying amount of aerodynamic blade sections are considered. First, the figure illustrates that the results again converge with an increased amount of employed aerodynamic sections. Second, by employing a reasonable amount of at least 30 sections it can be seen that during a relatively long simulation period of 7 seconds the energy varies with only 0.02%. The negative slope of this energy variation indicates that the respective amount of energy is pushed from the fluid to the structure. This very low energy flux seems to be rather uncritical for the present aero-elastic computations and it seems that the different energy levels of the two connected solvers can coexist relatively independent and undisturbed.
The figure also includes the energy variation of the loosely coupled simulation with integrated force scaling. As expected the respective energy variation is consistently equal to zero.

In the discussion above it is argued that the observed energy gap in Figure 7.5a between structure and fluid is not caused by the non-conservative force transfer but by the general differences in the respective wind turbine models. It is thus concluded here that the suggested force scaling of Section 5.3.1 is not recommendable for the present FSI coupling since it intents to level out the different energy levels of fluid and structural solver although those energies are most likely incomparable. It is further demonstrated in Figure 7.7b that a loosely coupled FSI simulation without force scaling is only pushing a very limited amount of erroneous energy from fluid to structure. This leads to the conclusion that the suggested force scaling is not necessary for sufficiently stable computations and that it is thus no longer taken into consideration for the simulations of the present work.

At the end of this section it is discussed whether a strong coupling should be considered for the subsequent FSI simulations of this chapter. Therefore, the energy variation between the energies $\Delta E_{S,FS}{}^{*}$ and $\Delta E_{S,FS}$ during a loosely coupled simulation without force scaling is calculated and illustrated in Figure 7.8. The computed quantity represents a measure for the inherent energy error of the employed loose coupling scheme and is easily comparable to the previously dicussed energy variation of Figure 7.7b. By comparing the two quantities, it can be seen that the inherent energy error is by a factor of 10^3 smaller than the previously discussed energy variation caused by the different energy levels between fluid and structural solver. Employing a strong coupling scheme could omit the

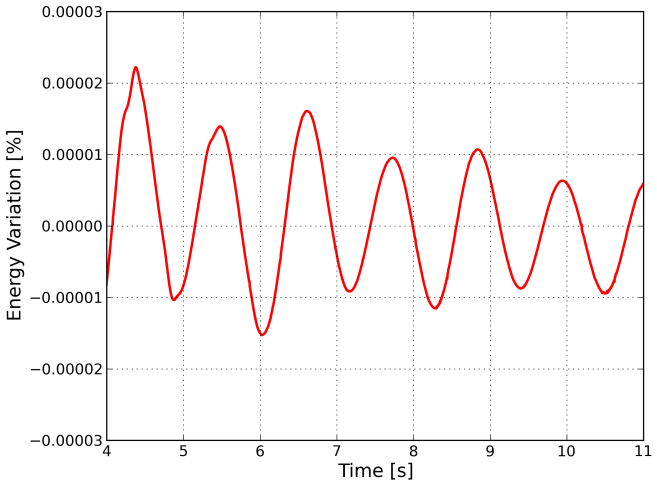


Figure 7.8: Inherent Erroneous Energy of Loose Coupling

inherent energy error, however, it is obvious that this partial improvement would not change the overall stability of the FSI simulation. Considering the decisive increase in computational costs as discussed in Section 7.2.2 it was decided that the strong coupling schemes are not further considered in the sequel of the present work. Generally, it can be stated here that due to the relatively small time step size of $\Delta t = 0.0025\,s$ the difference between predicted and actual deflection state is very small and that the inherent energy error of the loosely coupled algorithm is thus very small as well.

All future investigations and simulations will thus be carried out with the loosely coupled GSS scheme without force scaling as presented in Section 5.4.1. However, before employing the scheme extensively in the simulations of Section 7.3 and Section 7.4 it was desired to primarily assess the actual performance of the chosen coupling scheme in an elaborate time accuracy investigation. A good performance in the time accuracy investigations will at the same time also underline the sufficient numerical stability of the algorithm.

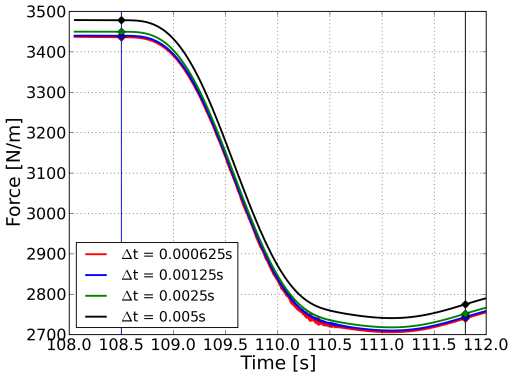
7.2.4 Time Accuracy of the loosely coupled GSS Scheme

Loosely coupled schemes were - and still are - frequently accused for being instable and reducing the order of time accuracy to the magnitude of only one. However, in Chapter 3 it was mentioned that in recent years progress was made in terms of the understanding and the improvement of loosely coupled systems. In the present work it was attempted to follow the suggestions summarized in Section

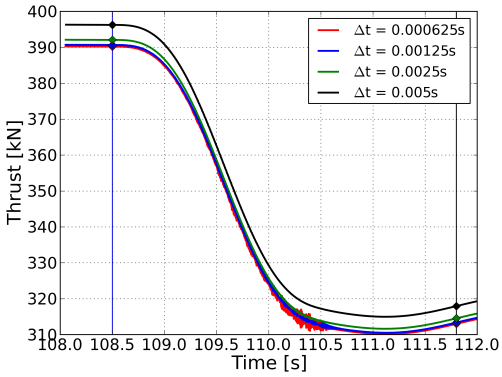
3.2 in order to establish a sufficiently stable loose coupling between HAWC2 and EllipSys3D which maintains the second order time accuracy of the stand-alone solvers. Considering the previous investigations in Section 7.2 it can be assumed that the relatively simple and computationally cheap loosely coupled GSS scheme of Section 5.4.1 can indeed provide sufficiently stable and sufficiently accurate results for the present aero-elastic computations, although the non-conservative force transfer could influence the expected performance. It was thus decided to better investigate this coupling scheme by conducting an elaborate time accuracy check for a substantial test case i.e. the aero-elastic simulation of an entire, flexible wind turbine during a sudden change in pitch angle.

Before checking the time accuracy of the overall FSI simulation tool we first focus on the time accuracy of the participating stand-alone solvers. Although the coupling scheme is expected to maintain the time accuracy of the stand-alone solvers it is not expected to improve the overall time accuracy above the primary limits.

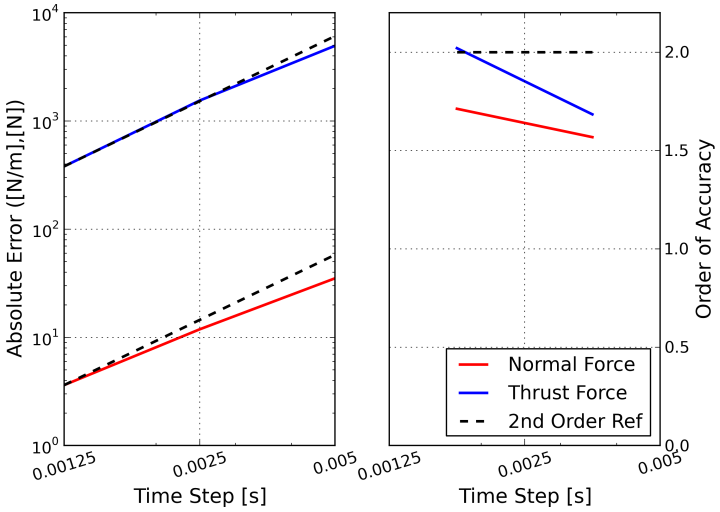
For the time accuracy investigation of the CFD code EllipSys3D we look at the aerodynamic forces during a prescribed step change in pitch angle from $0\,^\circ$ to $2\,^\circ$. It is the same test set-up as described in Section 7.2.1 but this time using a stiff structure of the NREL 5MW reference wind turbine. Both the rotation of the rotor and the pitching of the blades are achieved using the moving mesh method presented in Section 2.2.1. The present time accuracy investigation thus also checks the correct implementation of the theoretically second order time accurate moving mesh routines. In the present test case the second order space accurate SUDS scheme was used to project the convective velocities to the cell faces. The SUDS scheme was favoured over the third order space accurate QUICK scheme since a second order interpolation was better able to dampen out the high gradients in the flow field around the inner blade section with highly separated flow. This damping resulted in smoother force and thrust signals required for the subsequent time accuracy investigations.
In Figure 7.9a and Figure 7.9b the force and thrust curves of four simulations with four different time steps are shown, starting with a time step of $\Delta t = 0.005\,\mathrm{s}$ and continuing with halved time steps until a minimum time step of $\Delta t = 0.000625\,\mathrm{s}$. Assuming the simulation with the smallest time step of $\Delta t = 0.000625\,\mathrm{s}$ to be the exact solution we then calculate the absolute error of the other solutions by subtracting the respective results at a certain time t_{e2} from the reference value.



(a) Normal Force in $BLCS_F$ at $r = 70\,\%$ ($t_{e1} = 108.5\,s$, $t_{e2} = 111.8\,s$)



(b) Thrust Force of entire Wind Turbine ($t_{e1} = 108.5\,s$, $t_{e2} = 111.8\,s$)



(c) Absolute Error

Figure 7.9: Time Accuracy of EllipSys3D, Aerodynamic Response during Prescribed Pitch Motion (Stiff Structure)

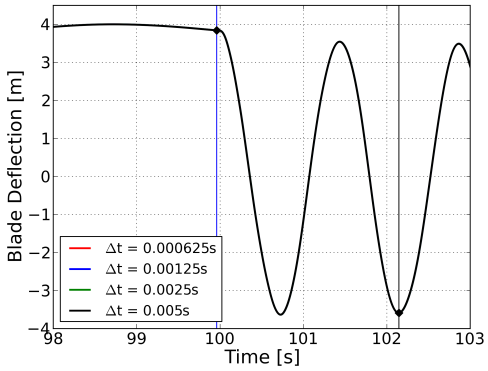
A second time incident t_{e1} is also marked in the plots, it is the time incident just before the prescribed pitch motion starts. The values at t_{e1} are used to nullify the error introduced to the computations during $t < t_{e1}$ and allows to focus solely on the error introduced by the prescribed pitch motion. The resulting absolute errors for $\Delta t = 0.00125\,\mathrm{s}$, $\Delta t = 0.0025\,\mathrm{s}$ and $\Delta t = 0.005\,\mathrm{s}$ of both the normal force in $BLCS_f$ at the radial blade position $r = 70\,\%$ and the integrated thrust force of the entire wind turbine are shown in Figure 7.9. We see that the calculated order of time accuracy is close to the theoretically expected second order time accuracy of the 3-point backward time integration scheme and the second order time accurate moving mesh method. The results show a time accuracy in

the range between the order of 1.6 and 2.0 which is a very satisfactory result when considering the fact that time accuracy investigation in practice are very delicate to conduct and that the present investigation was carried out on a full rotor simulation with very complex flow regimes.
As a final remark we here address the slightly noisy force signal of the simulation carried out with a time step size of $\Delta t = 0.000625\,\mathrm{s}$. This noise stems from the well-known problem of the Rhie-Chow interpolation scheme [62] used to omit the even-odd pressure decoupling while using collocated grid methods. In [76] it was found that the classical Rhie-Chow interpolation triggers a non-physical behaviour of the pressure if very small time steps are used. In the same work an improved interpolation method is introduced which eliminates the problem. However, the proposed method cannot be directly implemented in a moving mesh context and is thus not available for the present computations.

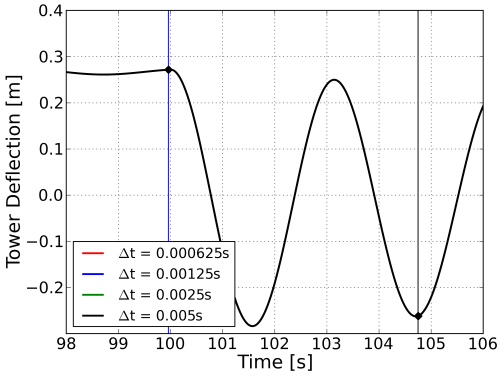
For the time accuracy investigation of the structural solver HAWC2 the NREL 5MW Reference turbine was again modelled during a constant and uniform inflow with a wind speed of of $U_\infty = 8\,m/s$, a pitch angle of $\Theta = 0\,^\circ$ and a constant rotational speed of $\omega = 0.968\,\mathrm{rad/s}$. In order to check the time accuracy of the structural solver the aerodynamic forces were instantaneously removed at the time t_{e1} and the structural response of the flapwise blade deflection as well as the fore-aft tower motion of the tower top were investigated. As done for the time accuracy investigation of EllipSys3D the computations were carried out for the four time steps $\Delta t = 0.000625\,\mathrm{s}$, $\Delta t = 0.00125\,\mathrm{s}$, $\Delta t = 0.0025\,\mathrm{s}$ and $\Delta t = 0.005\,\mathrm{s}$. In order to suppress disturbing frequencies in the structural responses the investigations considering the blade deflection were carried out with a stiff tower and the investigations of the tower top motion were carried out with stiff turbine blades. The two independent investigations of Figure 7.10a and Figure 7.10b both result in a very similar time accuracy of 1.4 to 1.7. This is slightly below the expected and theoretical time accuracy of the order of 2. Reasons could be found in the treatment of the external forces like gravity or in the general difficulty of determining the time accuracy in such a complex test case.

Now, the focus can be finally put on the results of the fully coupled system where the considered test case is the same as for the time accuracy investigations for the stand-alone solver EllipSys3D. However, in this final investigation the turbine structure is fully flexible and its structural response is now calculated by the structural solver of HAWC2. The prescribed pitch motion starts immedi-

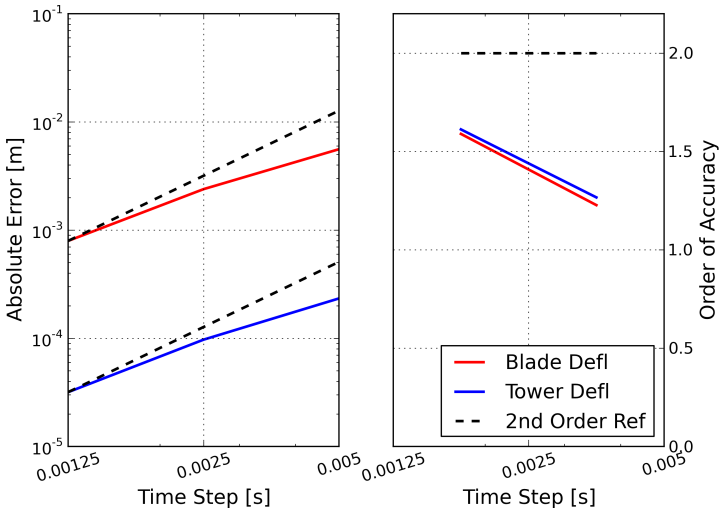
ately after the time instant t_{e1} and the resulting aero-elastic response can be seen in Figure 7.11a and Figure 7.11b. As before, the simulations have been carried out using the four time steps $\Delta t = 0.000625\,s$, $\Delta t = 0.00125\,s$, $\Delta t = 0.0025\,s$ and $\Delta t = 0.005\,s$ and by assuming the solution of the smallest time step to be exact the absolute errors of the other computations are determined and shown on the left side of Figure 7.11c. Both observed quantities the normal force at the blade position $r = 75\,\%$ and the flapwise blade deflection at $r = 95\,\%$ indicate that the overall time accuracy of the FSI coupling between HAWC2 and EllipSys3D is in the order between 1.5 and 2. The simple and computationally cheap loosely coupled GSS scheme of Section 5.4.1 is thus not reducing the time accuracy of the original stand-alone solvers and is indeed capable to reach a second order time accuracy. The investigation demonstrates that within the field of aeroelasticity a well designed loose coupling scheme can indeed provide results of higher order time accuracy and sufficient numerical stability. It is concluded that the investigated coupling scheme is well suited for the aero-elastic simulations of wind turbines and it is thus employed in all subsequent FSI computations of the present work.



(a) Flapwise Blade Deflection in $BLCS_S$ at $r = 95\,\%$
($t_{e1} = 99.95\,s$, $t_{e2} = 102.15\,s$)

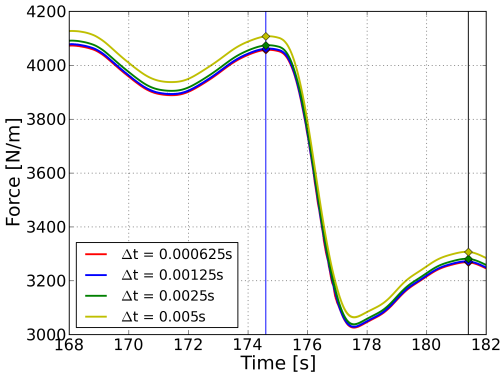


(b) Fore-Aft Tower Deflection at Tower Top
($t_{e1} = 99.95\,s$, $t_{e2} = 104.75\,s$)

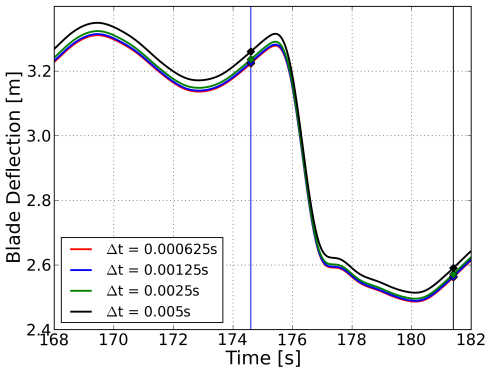


(c) Absolute Error

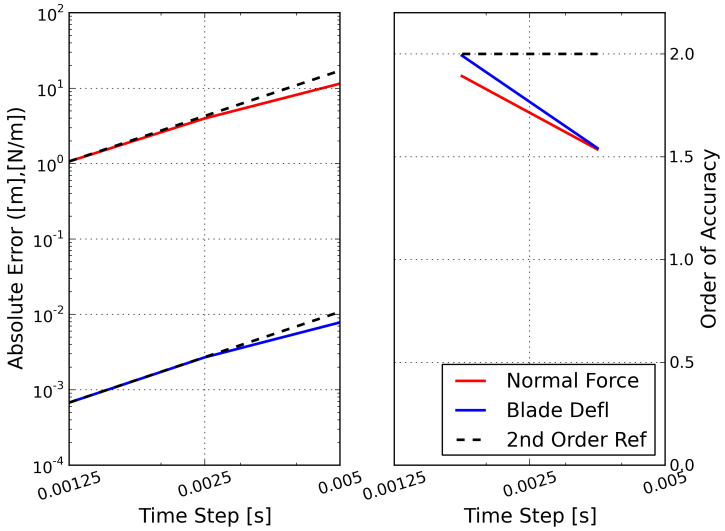
Figure 7.10: Time Accuracy of HAWC2, Structural Response after Removal of Aerodynamic Loads (Flexible Structure)



(a) Normal Force in $BSCS_F$ at $r = 75\,\%$ ($t_{e1} = 174.6\,s$, $t_{e2} = 181.4\,s$)



(b) Flapwise Blade Deflection in $BLCS_S$ at $r = 95\,\%$ ($t_{e1} = 174.6\,s$, $t_{e2} = 181.4\,s$)



(c) Absolute Error

Figure 7.11: Time Accuracy of FSI Simulation using HAWC2 and EllipSys3D, Aerodynamic and Structural Response during Prescribed Pitch Motion (Flexible Structure)

7.3 Power Curve and Related Quantities

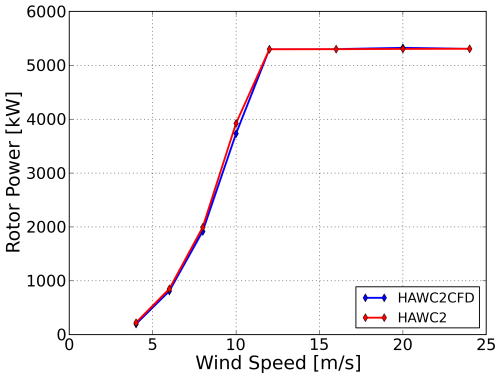
In the present section the developed high-fidelity FSI coupling between HAWC2 and EllipSys3D (HAWC2CFD) is used to compute the aero-servo-elastic response of the NREL 5MW reference turbine during different uniform inflow velocities. The investigations consider velocities in the range from 4 m/s to 24 m/s and thus cover the wind speeds in which a wind turbine is usually operating. The simulations can be considered as a basic test case in which the performance of the new FSI tool HAWC2CFD can be easily compared to the results of the traditional BEM-based model of HAWC2.

7.3.1 Rotor Integrated Quantities

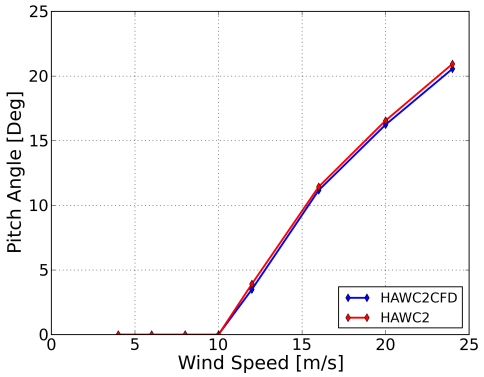
For every investigated wind speed the simulations are run until the aero-servo-elastic response converges to a quasi-steady state and in the graphs of Figure 7.12 every data point represents the converged result of one respective FSI computation.

The power curves of Figure 7.12a show that for wind speeds below rated power the computations of HAWC2CFD predict a slightly decreased power output when compared to the computations of HAWC2. This indicates that the turbine blades of the CFD model are aerodynamically slightly less efficient. For the wind speeds where rated power is reached the computed power outputs of HAWC2CFD and HAWC2 are nearly identical since the activated pitch controller enforces a constant power output by collectively pitching the blades. In Figure 7.12b it can be seen that the required pitch angles of HAWC2CFD are slightly smaller than the required angles of HAWC2. The smaller pitch angles lead to higher angles of attack and increase the aerodynamic forces of the HAWC2CFD model in order to achieve the same power output as in the comparable HAWC2 simulations. However, due to the higher angles of attack the thrust forces of HAWC2CFD are also increased. This can be seen in the thrust curves of Figure 7.12c. The rotor torque of Figure 7.12d shows the same characteristics as the previously discussed power curve.

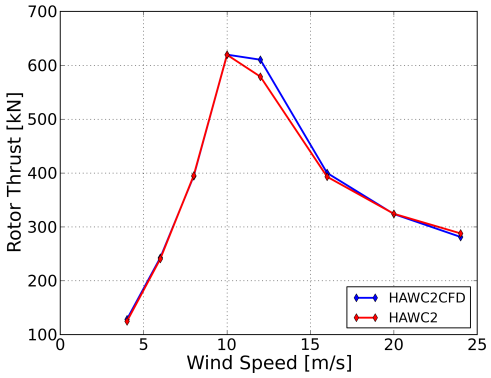
Below rated power the employed variable speed controller sets the generator torque in correspondence to the measured generator speed using a simple table look-up. In the HAWC2CFD simulations the aerodynamically less efficient turbine blades lead to a reduced rotor torque and to a reduced generator speed. The variable speed controller intents to react on the declining generator speed by reducing the generator torque and thereby re-increasing the generator speed.



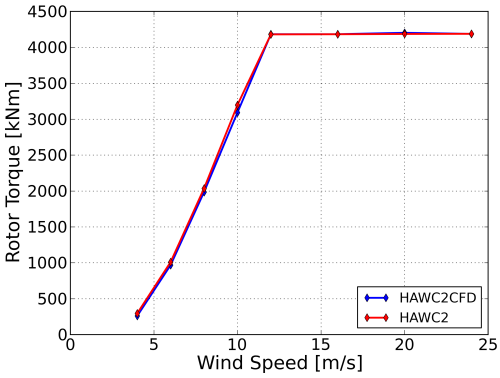
(a) Power Curve



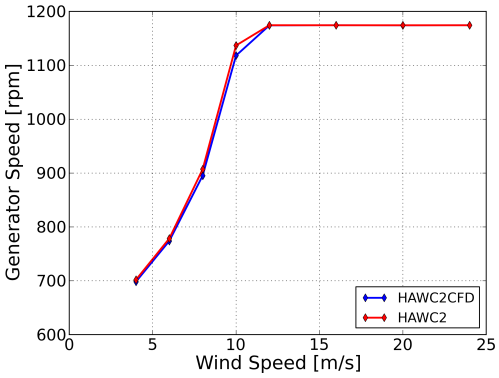
(b) Pitch Angles versus Wind Speed



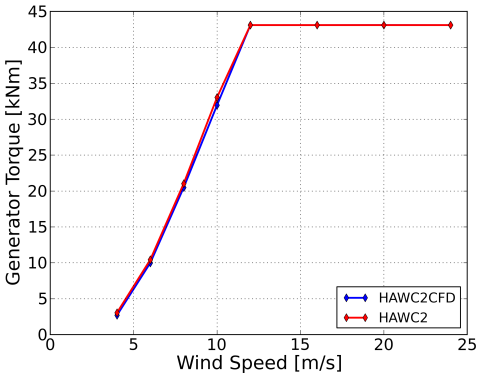
(c) Rotor Thrust versus Wind Speed



(d) Rotor Torque versus Wind Speed



(e) Generator Speed versus Wind Speed



(f) Generator Torque versus Wind Speed

Figure 7.12: Power Curve and Related Quantities computed with HAWC2 and HAWC2CFD

However, as seen in Figure 7.12e and Figure 7.12f, the found equilibrium between generator speed and generator torque is still slightly below the comparable values of the stand-alone HAWC2 computations.
Apart from the small discrepancies mentioned above, it can be generally stated that the simulation results of HAWC2CFD and of HAWC2 are in very good agreement.

7.3.2 Blade Force Distributions and Blade Deflections

In the previous section the full rotor FSI simulations of HAWC2 and HAWC2CFD showed very good agreements with respect to the integrated quantities such as power output, rotor torque and rotor thrust. In the present section the same computations are considered and compared, however, this time with the focus on the detailed force distributions over the wind turbine blades. Additionally, the calculated blade deflections are compared.

In Figure 7.13 to Figure 7.20 the distributed normal and tangential forces as well as the moments around the half chord axis are shown for all investigated wind velocities. The attributes *normal* and *tangential* are given with respect to the rotor plane.
At the root of the blade the tangential forces of HAWC2CFD are generally higher than the respective forces of HAWC2. While the tangential forces of HAWC2 increase rapidly at a relative blade position of $r/R \approx 10\,\%$ the forces of HAWC2CFD increase rather smoothly. Below the mentioned blade position the BEM model assumes a circular blade profile which only contributes to the aerodynamic forces with a certain drag force. The lifting force is assumed to be zero. However, since the complex three-dimensional flow patterns at the root of the blade are much better modelled with CFD, it is believed that the smooth increase in the HAWC2CFD simulations corresponds better to the real case.
Closer to the blade tip and for wind speeds below rated power the HAWC2 simulations show a slightly higher tangential force when compared to HAWC2CFD. The discrepancies could be connected with the implemented three-dimensional tip corrections in HAWC2 which might not be sufficiently conservative. The higher tangential forces at the blade tip are considered to be the reason for the increased rotor torque and rotor power of Figure 7.12. For wind speeds above 12 m/s the tangential forces at the blade tip become smaller than the respective forces of HAWC2CFD. This relative decrease can be explained with the higher

pitch angles in the HAWC2 simulations.

Close to the root the characteristics of the normal forces are very similar to the characteristics of the previously discussed tangential forces. While the normal forces of HAWC2CFD increase smoothly the respective forces of HAWC2 show a distinct increase at the relative blade position of $r/R \approx 10\,\%$. At relative blade positions with $r/R > 70\,\%$ the normal forces of HAWC2CFD are constantly higher than in HAWC2. For wind speeds below rated power this difference could be explained by the different aerodynamic inputs of the compared models. For higher wind speeds the difference is even more distinct. The reason can be found in the higher angles of attack of the HAWC2CFD computations resulting from the slightly reduced pitch angles shown in Figure 7.12b.

For wind speeds below rated power the computed moments of the HAWC2CFD computations are between 10 % and 30 % higher than the moments computed with HAWC2. For higher wind speeds the differences become relatively small.

The respective flapwise, edgewise and torsional blade deflections of the conducted FSI simulations are shown in Figure 7.21 to Figure 7.28 and the presented graphs demonstrate how well the predicted deflections of HAWC2 and HAWC2CFD agree with each other. The only distinct difference can be seen in the flapwise deflections for higher wind speeds where, due to the previously discussed increased normal forces towards the blade tip, the computations of HAWC2CFD show higher blade deflection.

The good agreement of the presented results demonstrates that the BEM-based aerodynamic model of HAWC2 is very well suited to accurately describe the underlying aerodynamic effects of the conducted test cases. The discussed differences in the aerodynamic forces at blade root and blade tip are rather small and have nearly no noticeable impact on the computed deflections. However, the aero-servo-elastic computations of the present section are quasi-steady and involve no distinct dynamic effects. Additionally, the pitch-to-feather mechanism assures that the turbine blades are constantly operating in attached flow regimes. In those rather simple test cases it is assumed that the BEM-based solver HAWC2 gives reliable results which then also compare well to the high-fidelity FSI computations of HAWC2CFD. In the subsequent section the rather complex test case of an emergency shut-down is investigated where the highly dynamic and complex

flow regimes are much more challenging for the BEM-based aerodynamic model of HAWC2.

Remark:
The normal forces and the flapwise deflection of Figure 7.3c and Figure 7.3d are not fully comparable to the respective forces and deflections of Figure 7.15 and Figure 7.23. In the comparisons of Section 7.2.1 the generator-torque controller was not activated and both the simulations of HAWC2 and the simulations of HAWC2CFD have been carried out with exactly the same rotational speed. In the comparisons of the present section, however, the rotational speed was determined by the aerodynamic forces and influenced by the activated generator-torque controller. For the comparison at 8 m/s this means that the HAWC2CFD simulations have been carried out with a slightly reduced rotational speed and that the computed forces and deflections are thus reduced respectively.

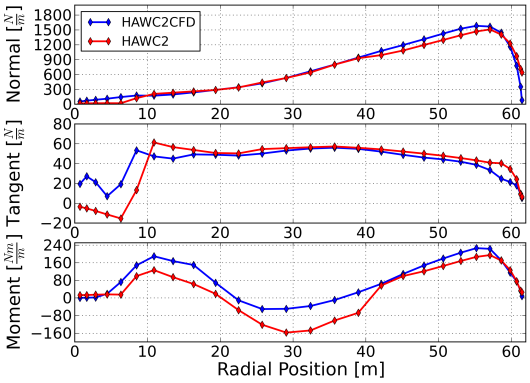


Figure 7.13: Normal Forces, Tangential Forces and Moments at 4 m/s

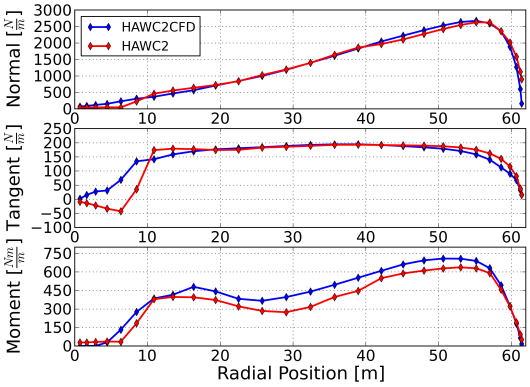


Figure 7.14: Normal Forces, Tangential Forces and Moments at 6 m/s

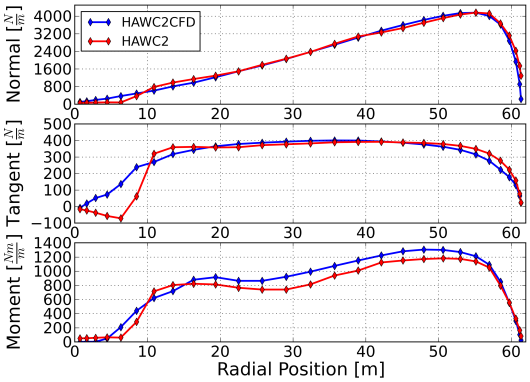


Figure 7.15: Normal Forces, Tangential Forces and Moments at 8 m/s

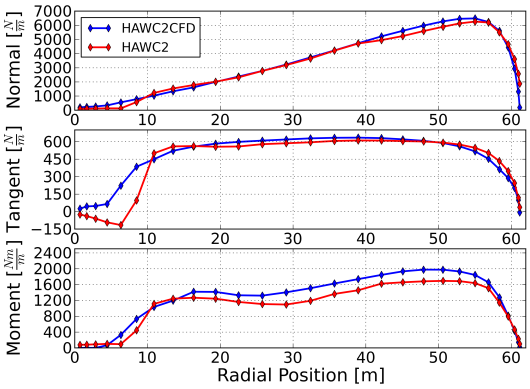


Figure 7.16: Normal Forces, Tangential Forces and Moments at 10 m/s

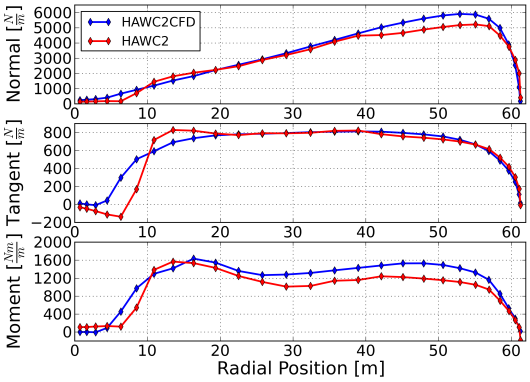


Figure 7.17: Normal Forces, Tangential Forces and Moments at 12 m/s

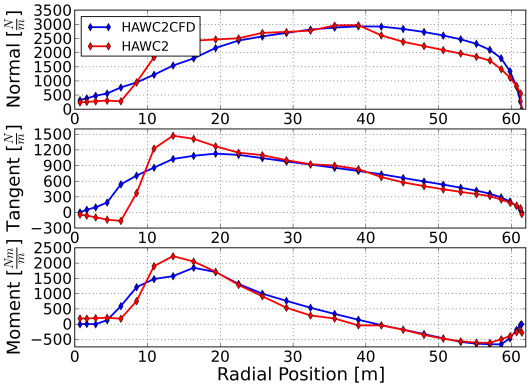


Figure 7.18: Normal Forces, Tangential Forces and Moments at 16 m/s

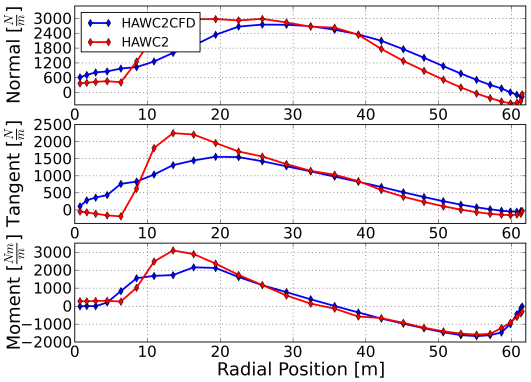


Figure 7.19: Normal Forces, Tangential Forces and Moments at 20 m/s

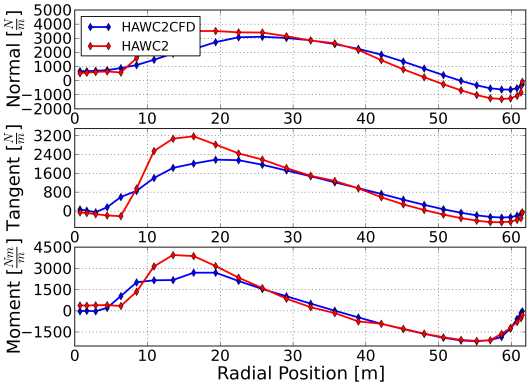


Figure 7.20: Normal Forces, Tangential Forces and Moments at 24 m/s

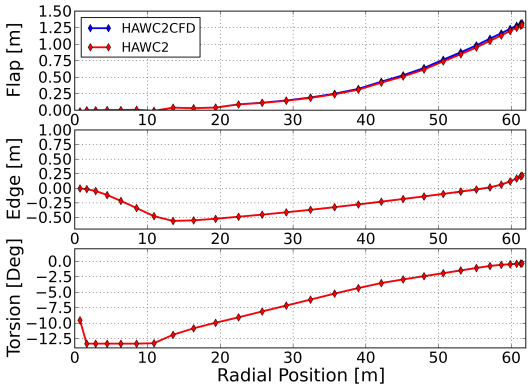


Figure 7.21: Flapwise, Edgewise and Torsional Blade Deflections at 4 m/s

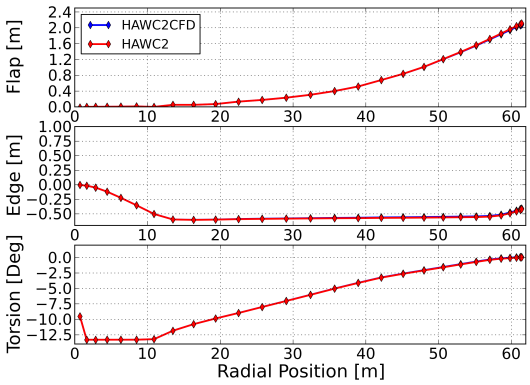


Figure 7.22: Flapwise, Edgewise and Torsional Blade Deflections at 6 m/s

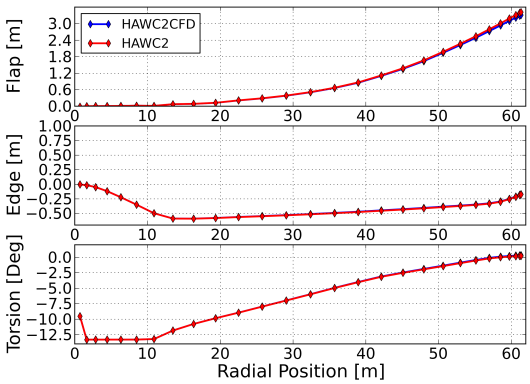


Figure 7.23: Flapwise, Edgewise and Torsional Blade Deflections at 8 m/s

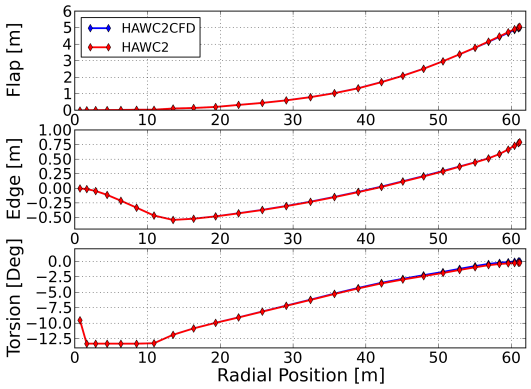


Figure 7.24: Flapwise, Edgewise and Torsional Blade Deflections at 10 m/s

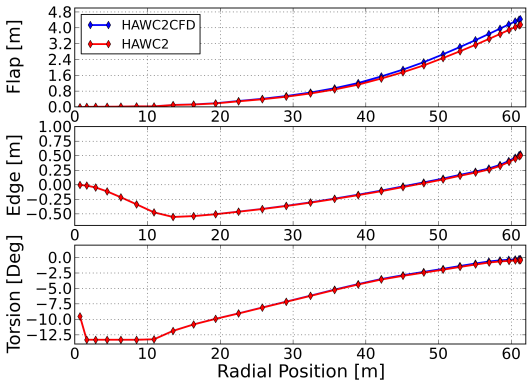


Figure 7.25: Flapwise, Edgewise and Torsional Blade Deflections at 12 m/s

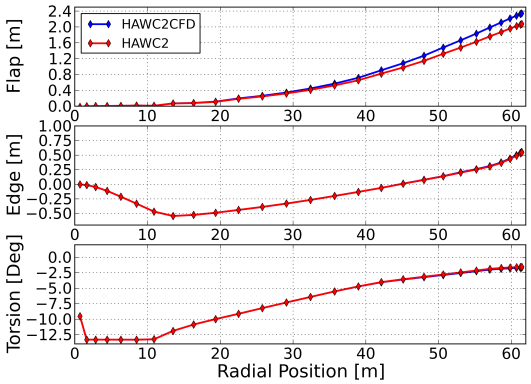


Figure 7.26: Flapwise, Edgewise and Torsional Blade Deflections at 16 m/s

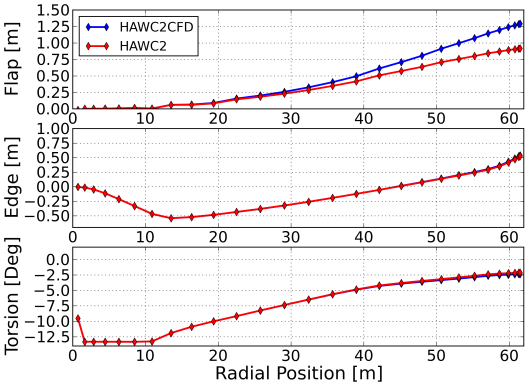


Figure 7.27: Flapwise, Edgewise and Torsional Blade Deflections at 20 m/s

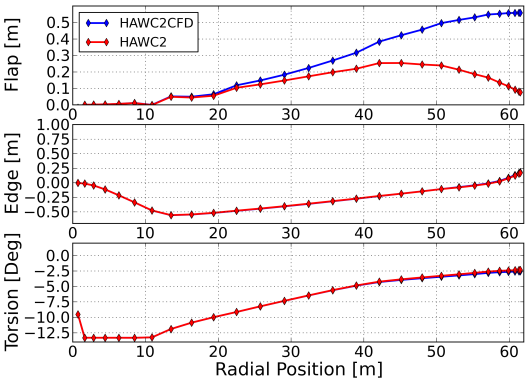


Figure 7.28: Flapwise, Edgewise and Torsional Blade Deflections at 24 m/s

7.3.3 Integrated Quantities during Switching of Aerodynamic Models

A final investigation of the present test cases considers the switching process from the computationally cheap BEM model to the computationally expensive CFD model. As explained in Appendix B.1 the switching of the aerodynamic models is used in the FSI simulations of HAWC2CFD in order to save computational time during the simulation of the initial transients. The switching takes place at the time instant $t = 167\,\mathrm{s}$.

Figure 7.29 shows the switching from HAWC2 to HAWC2CFD at a wind speed of $8\,\mathrm{m/s}$. Since the aerodynamic efficiency of HAWC2CFD is slightly reduced and since the pitch angle is held constant below rated power, the rotor torque drops slightly after the switching. Due to the reduced torque the rotor speed drops as well, however, the inertia of rotor and drive-train dampen the decrease. The generator torque is set in accordance to the measured generator speed and the new equilibrium is slightly below the initial level of the traditional HAWC2 computations. The disturbances during the switching process are relatively small, however, a full convergence of rotor torque and rotor speed takes longer than thirty seconds.

Figure 7.30 shows the switching from HAWC2 to HAWC2CFD at a wind speed of $24\,\mathrm{m/s}$. After the BEM-based forces are exchanged with the CFD-based forces the rotor torque starts to decrease since for the same pitch angles the loading predicted by HAWC2CFD is lower. As a result the generator speed drops with a certain delay. Since the generator speed is the input to the pitch control algorithm the controller reacts by pitching the blades in order to increase the aerodynamic forces. As a result the rotor torque recovers. Above rated speed the generator torque is set to be inversely proportional to the generator speed in order to keep the power output constant. After less than fifteen seconds the new equilibrium is found.

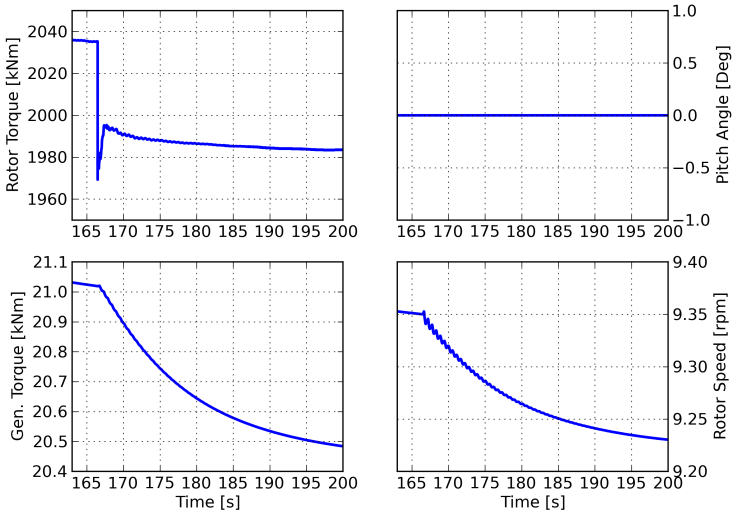


Figure 7.29: Switching from HAWC2 to HAWC2CFD at 8 m/s

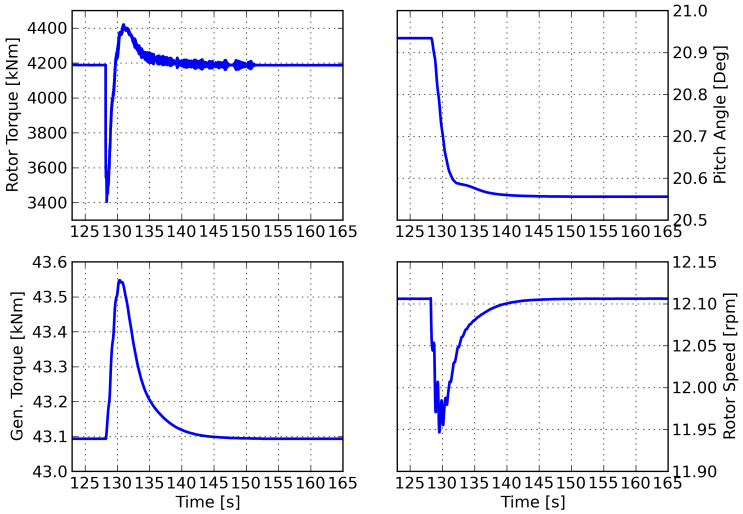


Figure 7.30: Switching from HAWC2 to HAWC2CFD at 24 m/s

7.4 Emergency Shut-down

In the simulations of Section 7.3 the NREL 5MW reference turbine was modelled during constant inflow velocities and the involved dynamic effects were rather small. The variable pitch-to-feather controller assured small angles of attack and attached flow regimes over most parts of the blades. In such conditions the BEM based aerodynamic model of the stand-alone HAWC2 solver was expected to give reliable results and the respective investigations of the previous section showed indeed a good agreement between the computationally cheap FSI simulations of HAWC2 and the high-fidelity FSI simulation of HAWC2CFD.

In the present section the NREL 5MW turbine is modelled during an emergency shut-down after a sudden power loss. The loss in power results in a sudden loss in generator torque and action has to be taken in order to prevent the wind turbine from spinning faster and faster. In such a case of emergency the wind turbine has to be slowed down aerodynamically by quickly pitching the blades towards smaller angles of attack.
The fast pitching motion changes the aerodynamic loading on the rotor rapidly and severely, and the modelling of such a highly dynamic change in aerodynamics could be a considerable challenge for the BEM based aerodynamic model of HAWC2. Additionally, the rapid change in the aerodynamic loading provokes a very strong structural response in which the wind turbine blades are subjected to rather unusual and complex flow regimes. The simulation of an emergency shut-down is therefore considered to be a very complex and thus a very good test case in order to compare the results of the traditional stand-alone solver HAWC2 with the high-fidelity FSI computations of HAWC2CFD.

The results of this section consider an emergency shut-down at a wind speed of 10 m/s where, as seen in the thrust curve of Figure 7.12c, the aerodynamic loading is highest. Since the emergency shut-down will rapidly reduce the aerodynamic loading to zero the respective aero-elastic response is expected to be most extreme at this particular wind speed.
Figure 7.31 shows several quantities of the simulated emergency shut-down. In the considered simulation case the loss of generator torque occurs at the time $t = 145\,\mathrm{s}$. It is assumed that the control algorithm needs the time $t_{delay} = 0.2\,\mathrm{s}$ in order to detect the loss of generator torque and to initialize the required pitch motion. The controller is then pitching the blades with a maximum pitch velocity

of $\dot{\Theta} = 10\,^{\circ}/\mathrm{s}$ until the maximum pitch angle of $\Theta = 90\,^{\circ}$ is reached. In Figure 7.31b the respective pitch angles are plotted over time.
The pitch motion decreases the aerodynamic loading significantly and it can be seen in both the power curve of Figure 7.31a, the rotor thrust curve of Figure 7.31c and the rotor torque curve of Figure 7.31d that the rapid change of the pitch angle is temporarily changing the wind turbine generator into a fan of reversed power, reversed thrust and reversed torque values. The negative power output of the rotor slows the wind turbine quickly down and by observing the rotor speed of Figure 7.31e it can be seen that within fifteen seconds the turbine is nearly completely stopped. Figure 7.31f shows that the sudden decrease of the rotor thrust also triggers a strong fore-aft motion of the wind turbine tower top.

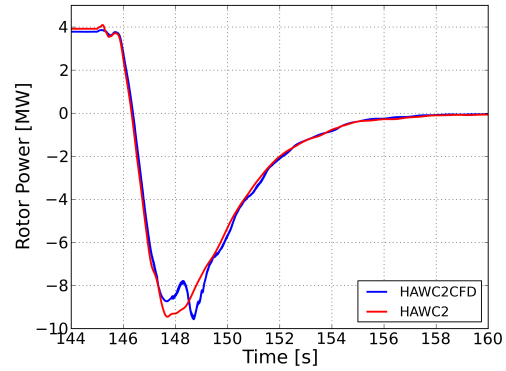
A comparison between the results of HAWC2 and HAWC2CFD reveals that both simulations predict a quite similar aero-elastic response. This is a remarkable result and demonstrates that the low-fidelity aerodynamic modelling of HAWC2 is still capable to sufficiently describe the complex aerodynamic effects of an emergency shut-down. Some discrepancies between the models can be observed in the computed power and torque curves of Figure 7.31a and Figure 7.31d where the computations of HAWC2CFD show a distinct disturbance at $t \approx 149\,s$. However, the observed differences do not seem to have a particular impact on the overall characteristics of the computed aero-elastic response.

The comparison between HAWC2 and HAWC2CFD is pursued by observing the computed blade forces and blade deflections. In Figure 7.32, Figure 7.33 and Figure 7.34 the sectional forces at the four radial blade positions of $r_{SEC=10} = 13.5\,\mathrm{m}$, $r_{SEC=15} = 29.0\,\mathrm{m}$, $r_{SEC=20} = 45.1\,\mathrm{m}$ and $r_{SEC=25} = 57.0\,\mathrm{m}$ are plotted over time.[1]
It can be seen in Figure 7.32 that the forces normal to the observed turbine blade are in a formidable agreement. However, Figure 7.33 illustrates that the forces tangential to the turbine blade differ much more from each other. At the outer blade sections the tangential forces of HAWC2CFD vary with an increased amplitude of more than 50% when compared to the simulations of HAWC2. This amplified variation in the tangential forces could already be noticed in the previously discussed power and torque curves of Figure 7.31a and Figure 7.31d. Figure 7.34 shows that the computed moments of HAWC2CFD are also slightly increased.

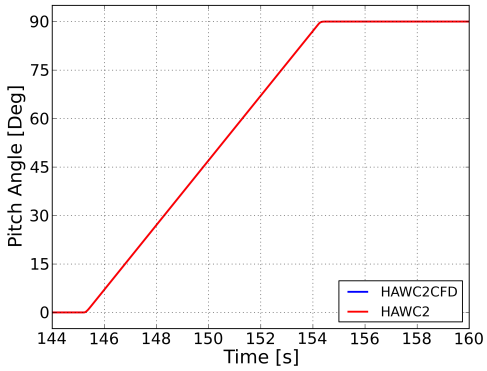
[1]The sectional number indicates the number of the considered blade section starting with Section 1 at the blade root

The respective flapwise, edgewise and torsional blade deflections are shown in Figure 7.35, Figure 7.36 and Figure 7.37. Apart from a slight phase shift at the end of the simulated test case the computed blade deflections of HAWC2 and HAWC2CFD are very similar. The observed differences in the aerodynamic forces do not seem to have any major impact on the resulting structural deflections.

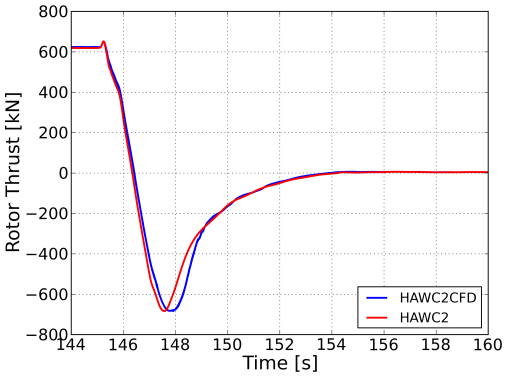
In the previous discussion it is pointed out that both models the BEM-based solver HAWC2 and the CFD-based solver HAWC2CFD predict a very comparable aero-elastic response for the simulated emergency shut-down. However, apart from the above compared quantities the high-fidelity FSI simulation of HAWC2CFD can give additional information about quantities that are not accessible in a traditional HAWC2 simulation. As an example for such a quantity the time dependent pressure distributions at the four previously mentioned blade sections are shown in Figure 7.38 to Figure 7.43. The pressure distributions at the time instant $t = 145.0\,\mathrm{s}$ represent the initial situation before the loss of power occurs. The respective distributions look like regular pressure distributions where the stagnation points with a pressure coefficient of $C_P = 1$ are located on the lower side of the aerofoil sections. The highest negative pressure gradient and thus the highest risk of flow separation is found at the innermost section of radius $r_{SEC=10} = 13.5\,\mathrm{m}$. At the subsequent time instant $t = 145.9\,\mathrm{s}$ the pitch angle has an approximate value of $\Theta = 5\,^\circ$ and due to the reduced angles of attack the lift forces are decreased as well. This can be seen in the respective snapshots of Figure 7.39 where the enclosed areas of the pressure distributions are decreased and the minimum values of the pressure coefficients are increased. At the time instant $t = 146.8\,\mathrm{s}$ the pitch angle is further increased and the respective angles of attack start to become negative. Figure 7.40 reveals the negative angles of attack since the respective stagnation points with a pressure coefficient of $C_P = 1$ are now located on the upper sides of the aerofoil sections. During the last three snapshots of Figure 7.41, Figure 7.42 and Figure 7.43 it can be seen that the pressure distributions of the outer blade sections start to exhibit extremely negative pressure gradients at the area close to the leading edge which is then followed by an area of nearly constant pressure values. This clearly indicates that the flow is separating at the leading edge.



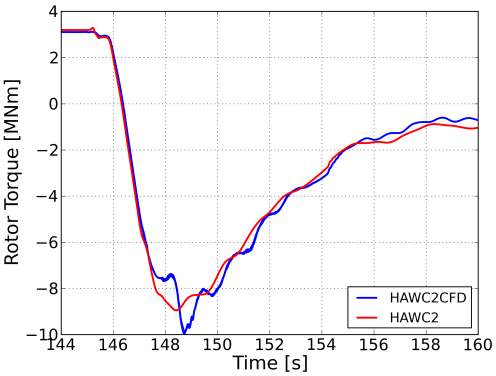
(a) Rotor Power Over Time



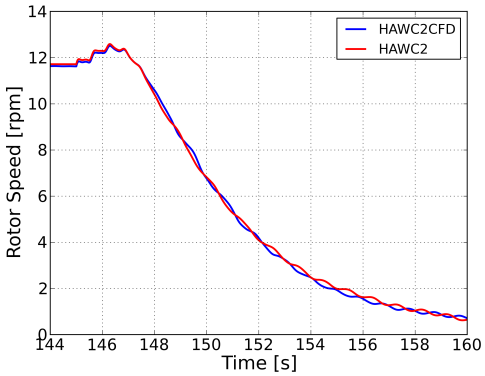
(b) Pitch Angle Over Time



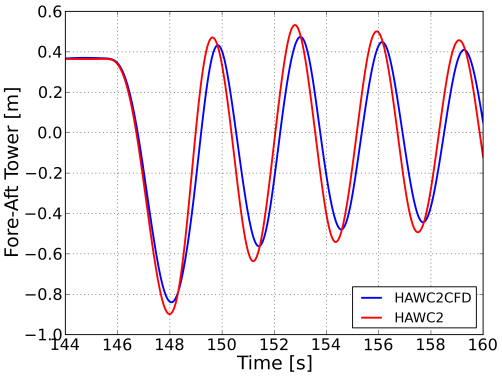
(c) Rotor Thrust Over Time



(d) Rotor Torque Over Time



(e) Rotor Speed Over Time



(f) Fore-Aft Tower Top Motion Over Time

Figure 7.31: Emergency Shutdown at 10 m/s computed with HAWC2 and HAWC2CFD

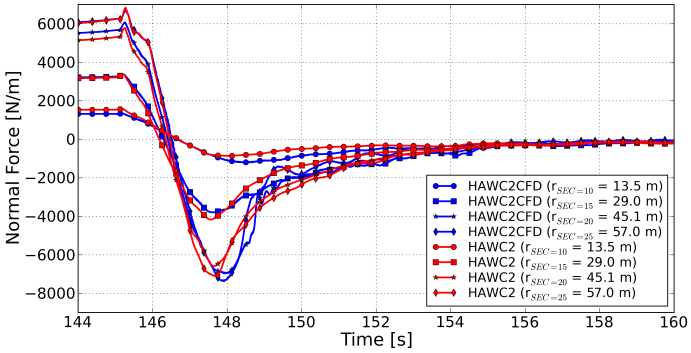


Figure 7.32: Emergency Shut-down at 10 m/s - Normal Force in $BLCS_S$ at Different Blade Sections

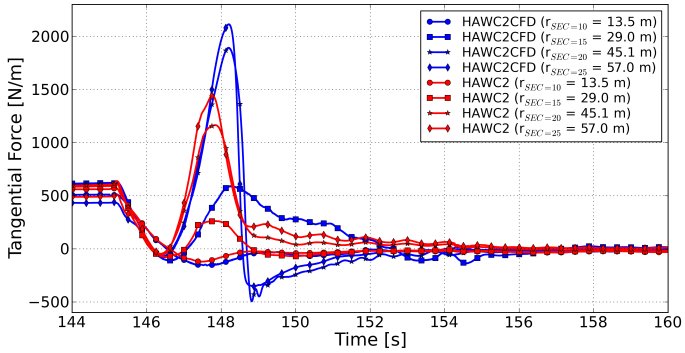


Figure 7.33: Emergency Shut-down at 10 m/s - Tangential Force in $BLCS_S$ at Different Blade Sections

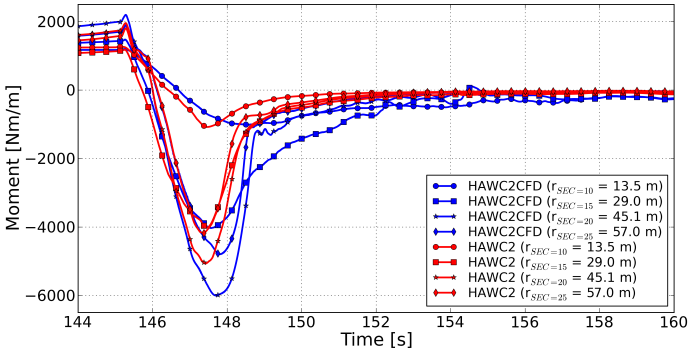


Figure 7.34: Emergency Shut-down at 10 m/s - Torsional Moment in $BLCS_S$ at Different Blade Sections

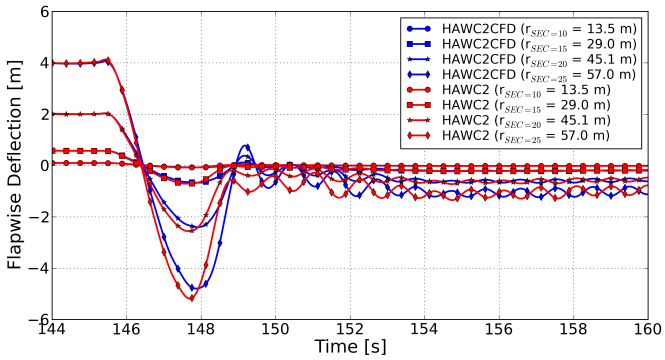


Figure 7.35: Emergency Shut-down at 10 m/s - Flapwise Deflection at Different Blade Sections

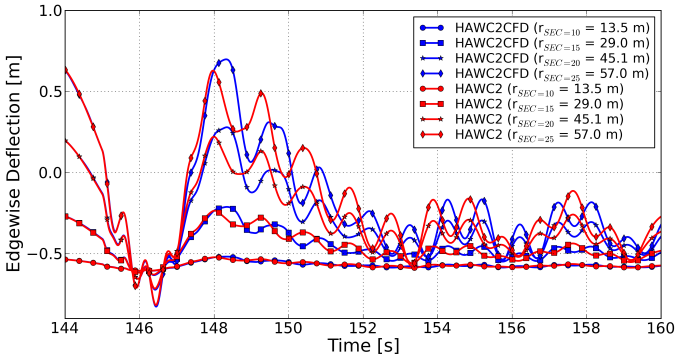


Figure 7.36: Emergency Shut-down at 10 m/s - Edgewise Deflection at Different Blade Sections

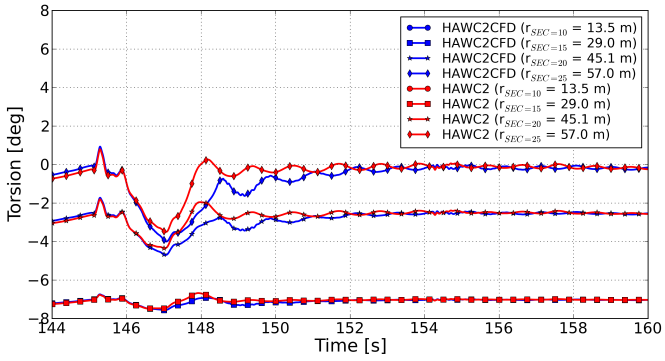


Figure 7.37: Emergency Shut-down at 10 m/s - Torsional Deflection at Different Blade Sections

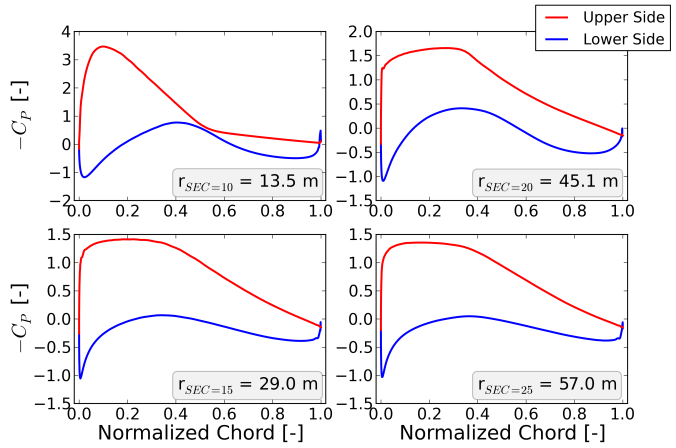


Figure 7.38: Emergency Shut-down at 10 m/s - Pressure Distributions at $t = 145.0\,s$

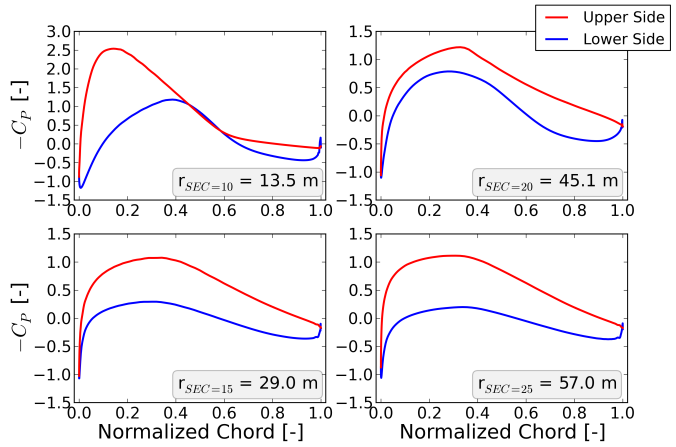


Figure 7.39: Emergency Shut-down at 10 m/s - Pressure Distributions at $t = 145.9\,s$

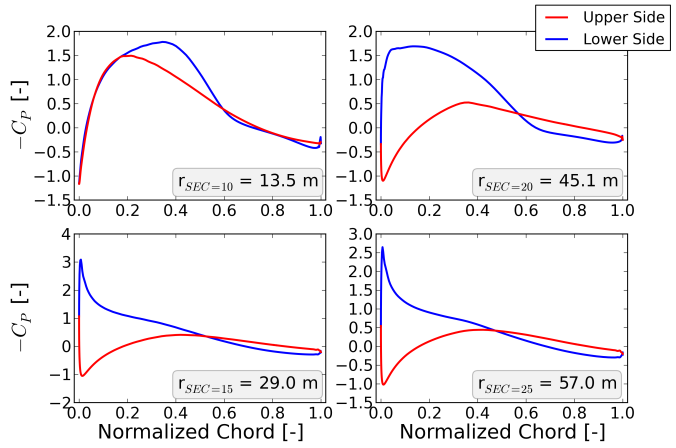


Figure 7.40: Emergency Shut-down at 10 m/s - Pressure Distributions at $t = 146.8\,s$

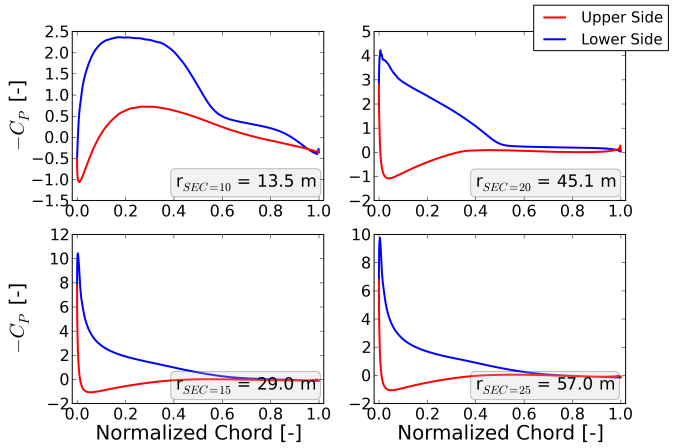


Figure 7.41: Emergency Shut-down at 10 m/s - Pressure Distributions at $t = 147.7\,s$

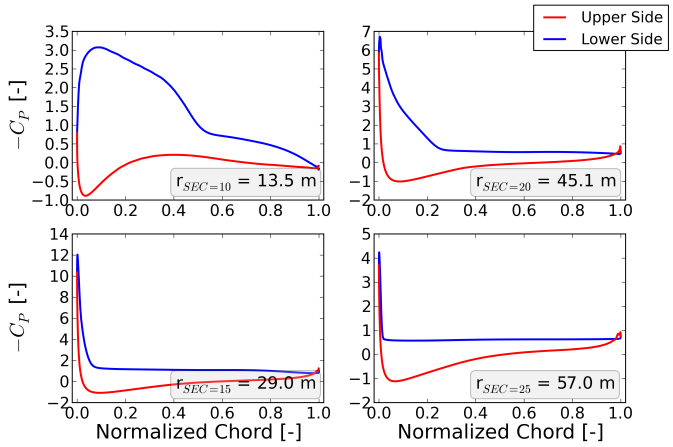


Figure 7.42: Emergency Shut-down at 10 m/s - Pressure Distributions at $t = 148.6\,s$

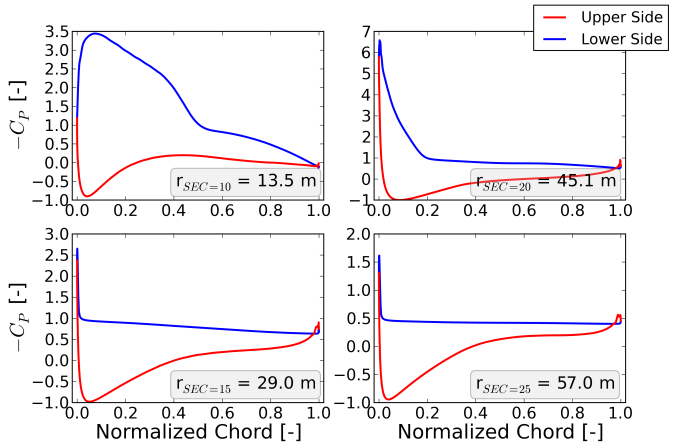


Figure 7.43: Emergency Shut-down at 10 m/s - Pressure Distributions at $t = 149.5\,s$

Chapter 8

Conclusion

The main focus of the present work was to establish a high-fidelity fluid-structure interaction (FSI) simulation tool for full rotor computations in wind energy. The work followed a partitioned coupling approach in which the multi-body-based structural model of the aero-elastic code HAWC2 was coupled with the finite volume Navier-Stokes solver EllipSys3D. Both stand-alone versions of the participating solvers have been developed at the predecessor departments of DTU Wind Energy and have been proven to provide reliable results in numerous applications. The partitioned coupling of the aero-elastic solver HAWC2 and the computational fluid dynamics (CFD) solver EllipSys3D constitutes one of the first high-fidelity FSI simulation tools in the field of wind energy.

The work discussed several ideas and suggestions for partitioned coupling approaches in the field of aeroelasticity and biomechanics and presented several key components that are needed in order to establish a loosely coupled FSI simulation of higher order time accuracy and sufficient numerical stability. The coupling between HAWC2 and EllipSys3D was then established by incorporating the previously discussed ideas and requirements into five different coupling schemes. The various schemes were implemented in order to examine the consequences of using either a loosely coupled or a strongly coupled scheme and of using either a non-conservative or a conservative force and deflection transfer.

The coupling schemes were then assessed in a full rotor FSI simulation of the NREL 5MW reference turbine. By monitoring the energies released and received by fluid and structure it could be shown that the implemented non-conservative force and deflection transfer has no noticeable effects on the numerical stability of the conducted FSI computations. It could also be shown that a computationally expensive strong coupling scheme is not required for the considered aero-elastic

simulations. A detailed time accuracy investigation could finally demonstrate that a loosely coupled generalized serial staggered (GSS) scheme with a non-conservative force and deflection transfer is capable of providing the desired second order accurate FSI simulations.
The developed FSI coupling between HAWC2 and EllipSys3D (HAWC2CFD) was then employed to conduct several aero-servo-elastic computations. In a first set of simulations the NREL 5MW wind turbine was modelled during different uniform wind velocities between 4 m/s and 24 m/s and several quantities like rotor power, rotor thrust, generator speed and generator torque as well as the blade force distributions and blade deflections were compared with the results of the BEM-based stand-alone version of HAWC2. Most of the small differences between the models could be explained with the aerodynamically less efficient turbine blade geometry in HAWC2CFD. Some other differences in the force distributions at the blade root and the blade tip could be explained with the inherent limitations of the BEM-based aerodynamics of HAWC2. However, the observed discrepancies were rather small and both models predicted very similar aero-servo-elastic responses. Finally, the rather complex test case of an emergency shut-down was simulated with both the BEM-based stand-alone solver of HAWC2 and the high-fidelity FSI simulation tool HAWC2CFD. Although some differences could be observed in the predicted tangential forces the results of the two models were still in a very good agreement.

Within the scope of this work a generic coupling framework was developed in Python in order to establish the partitioned FSI coupling between HAWC2 and EllipSys3D. The framework was developed in order to orchestrate the execution of the solvers and the data transfer between the solvers in a standardized way. The standardized formulation makes it possible to easily exchange the participating solvers and to employ the framework for other couplings as well. The coupling framework could thus also be used to couple a simple three degrees of freedom (3-DOF) structural solver with both the two-dimensional CFD solver EllipSys2D and the three-dimensional CFD solver EllipSys3D. The established coupling was employed in the work of Skrzypiński in order to investigate both stall-induced and vortex-induced vibrations of wind turbine blades during standstill. After connecting a simple control model to the framework the established aero-servo-elastic simulation tool could be used to investigate the load reduction potential of a two-dimensional aerofoil section equipped with trailing edge flaps.

8.1 Outlook

In the conducted test cases of the present work the results of the BEM-based aero-elastic solver HAWC2 compare very well with the results of the new high-fidelity FSI solver HAWC2CFD. In the quasi-steady computations during constant wind speeds the aerodynamic model of HAWC2 proved its good performance and predicted an aero-elastic response that agrees very well with the results of HAWC2CFD. During the simulation of the emergency shut-down the differences in the predicted tangential forces where rather big, however, in the particular case of an emergency shut-down the forces are only acting for a very short period of time and seem to have no significant impact on the respective structural response. In future investigations it is therefore desirable to consider test cases where the limits of the BEM-based aerodynamics are challenged but, additionally, where the predicted forces act longer or periodically on the investigated wind turbine structure. Conceivable test cases could be a three-dimensional full rotor investigation of both stall-induced and vortex-induced vibrations during stand-still or the investigation of flutter limits. Further test cases should comprise simulations with unsymmetrical rotor loadings during e.g. yawed, skewed and partial wake situations in order to investigate the performance and the eventual limits of the dynamic inflow model in HAWC2.

In case the future investigations show an increased demand on the numerical stability of the developed FSI simulation tool, it could be interesting to implement a conservative force and deflection transfer as outlined in Appendix B.4 in order to examine its influences on the results. For more delicate FSI investigations it would be also desirable to better understand the relatively big differences in the predicted blade moments of Section 7.3.2.

Appendix A

The Python Coupling Framework in Practice

A.1 File Structure and Examples

The Python coupling framework consists of a few files only. The core part of the framework is found in the main script *MainMPI.py* and in the file *cpl.py* which contains some supplemental functions in order to keep the main script clearly arranged. Apart from those two files the coupling framework needs to access some model specific functions which for each participating model are stored in one additional file. In case the coupling framework is connected with the fluid solver EllipSys3D this would mean that the additional file *cpl_el3d.py* needs to be available. Finally the input file *varlist.dat* presented in Section 4.4.2 is needed in order to organize the data transfer.

The present section shows the typical content of the files *MainMPI.py*, *cpl.py* and *cpl_el3d.py* and thereby gives an insight into the functionality, the simplicity and the lucidity of the developed framework.

The main script *MainMPI.py*

The main script *MainMPI.py* of the developed Python coupling framework is shown in Listing A.1. The program execution is divided into six stages.

- In Stage 1 the Python related initialization processes are carried out. As a first step the participating models are loaded into the script in order to make the standardized interface functions of Section 4.4 available for the Python script. Subsequently the supplemental functions (found in *cpl.py*)

and the model specific functions (e.g. found in *cpl_el3d.py*) are loaded into the script. As a next step the message passing interface for parallel computations *MPI4Py* is loaded and initialized. Then, several Python variables are initialized by e.g. reading the Python input file *varlist.dat*. Subsequently some logical variables are set in order to identify which models will be used for the current coupling. The variables determine the coupling mode of each connected model and finally decide which of the implemented coupling related functions will be employed inside the respective models.

- In Stage 2 the participating models are initialized and thus accessed for the first time. The interface function INIT() is called in order to read in the individual input files of each connected solver. Then, the desired output variables are read out and eventually broadcast to the other computational nodes. It should be noticed here that the serial codes are only called on node 1 (i.e. $rank = 0$). As mentioned in Section 4.4.2 the read-out of the variables is divided into a preparation step and an actual read-out step. The function calls related to the data flow handling refer to functions which are defined in the file *cpl.py*. They are not yet the actual interface functions which directly communicate with the connected solvers.

- Stage 3 represents the synchronization step which is here divided into the two steps SYN1 and SYN2. The calls of both the action control functions SYN1() and SYN2() are preceded by the respective input functions. In the present example it is only the synchronization step SYN1 that is succeeded by some respective output functions. Finally the input and the output for the subsequent STEP() functions are prepared.

- Stage 4 is used to manage and accelerate the start-up procedure of the coupling. Further information is given in Section A.3.

- Stage 5 is containing the actual time loop of the coupling. Examples for both a loose coupling and a strong coupling are explained below and shown in Listing A.2.

- Stage 6 is used to terminate the models in a proper way.

Listing A.1: Main script *MainMPI.py*

```
# _______
# stage 1: initialization of python framework

# load participating models (python wraps)
import hawc2 as hawc
import structure as strc
import flapcontrol as fctr
import ellipsys2D_MPI as el2d
import ellipsys3D_MPI as el3d

# load supplemental and model specific functions
import cpl
import cpl_hawc
import cpl_strc
import cpl_fctr
import cpl_el2d
import cpl_el3d

# load several python modules
from mpi4py import MPI

# initialize MPI4PY
comm = MPI.COMM_WORLD
rank = comm.Get_rank()

# load and organize python input
cpl.read_python_input()
cpl.create_tailored_lists()

# select participating models
HAWC, STRC, FCTR, EL2D, EL3D=cpl.select_models()

# determine coupling mode of models
if HAWC: cpl_hawc.cpl_mode(EL3D)
if STRC: cpl_strc.cpl_mode(EL2D,EL3D)
if STRC: cpl_fctr.cpl_mode(EL2D)
if EL2D: cpl_el2d.cpl_mode(FCTR,STRC)
if EL3D: cpl_el3d.cpl_mode(HAWC,STRC)

# _______
# stage 2: initialization of models

# initialize models with individual input
if rank == 0:
 if HAWC: hawc.init()
 if STRC: strc.init()
 if FCTR: fctr.init()
if EL2D: el2d.init()
if EL3D: el3d.init()

# fetch variables after INIT
cpl.prepare_init_output(rank)
cpl.init_output(rank)
cpl.init_broadcast(rank,comm)

# _______
# stage 3: synchronization of models

# run synchronization (step 1)
cpl.prepare_syn1_input(rank)
cpl.syn1_input(rank)
if rank == 0:
 if HAWC: hawc.syn1()
 if STRC: strc.syn1()
 if FCTR: fctr.syn1()
if EL2D: el2d.syn1()
if EL3D: el3d.syn1()

# fetch variables after SYN1
cpl.prepare_syn1_output(rank)
cpl.syn1_output(rank)
cpl.syn1_broadcast(rank,comm)

# run synchronization (step 2)
cpl.prepare_syn2_input(rank)
cpl.syn2_input(rank)
if rank == 0:
 if HAWC: hawc.syn2()
 if STRC: strc.syn2()
 if FCTR: fctr.syn2()
if EL2D: el2d.syn2()
if EL3D: el3d.syn2()

# prepare STEP variables / synchronize dimensions
cpl.prepare_step_output(rank)
cpl.step_broadcast_dim(rank,comm)
cpl.prepare_step_input(rank)

# _______
# stage 4: reduction of start up transients
cpl.reduce_transients(rank,comm)

# _______
# stage 5: time loop
...
(details given in Listing 4.2)
...

# ______
# stage 6: terminate models
if rank == 0:
 if STRC: strc.finalize()
if EL2D: el2d.finalize()
if EL3D: el3d.finalize()
```

The course of Stage 5 in the main script *MainMPI.py* depends on whether a loose or a strong coupling scheme is selected. Considering the loose coupling scheme on the right side of Listing A.2 the following three constellations are conceivable.

- In case the models of HAWC2 and EllipSys3D are activated we see that HAWC2 is first called with the action control function STEP_PREDICT() in order to predict the deflection state of the new time step. The predicted deflections are then read out by using the *step_output()* function defined in the model specific file *cpl_hawc.py*. This output is then available as an input to the STEP() function of EllipSys3D where the predicted deflection state is used to calculate the respective aerodynamic forces. Those forces are subsequently fetched as output and then given as input to HAWC2 where they are used to calculate the corrected deflection state within the interface function STEP_CORRECT(). A schematic representation of this loosely coupled scheme was given in Figure 3.1.

- In case the 3-DOF structural model STRC is coupled with the two-dimensional or three-dimensional version of EllipSys3D, the time loop starts by calling the STEP() function of the structural solver and calculates its solution of

the new time step purely explicit by using the input values (e.g. the aerodynamic forces from EL2D or EL3D) from the previous time step. This new deflection is then used as input for the STEP() function of EllipSys where the aerodynamic forces of the current time step are calculated. The structural model employs a simple and fully explicit version of the Runge-Kutta-Nyström time integration scheme suggested in Øye [70]. A schematic representation of this elementary loose coupling was given in Figure 3.2a.

- In order to accomplish the two-dimensional aero-servo-elastic computations of Heinz [55] the 3-DOF structural model STRC is coupled with EllipSys2D and with the flap control model FCTR. The aero-elastic coupling works in the same way as explained in the previous item. The flap controller is then using the flowfield of the current time step as control input and provides an updated flap position to EllipSys2D in order to update the CFD mesh for the next time step.

On the left side of Listing A.2 the time loop of a strong coupling scheme is given. Since the strong coupling is only implemented for the coupling between HAWC2 and EllipSys3D no other models appear within this loop.

- For the strong coupling between HAWC2 and EllipSys3D the outer time loop now also includes an additional subiteration loop where the two models are called alternately in order to exchange the aerodynamic forces and the structural deflections at each subiteration. In the current implementation the subiteration loop is exited as soon as the convergence criteria of HAWC2 is met or in case a maximum number of fluid solver subiterations is reached. The three action control functions STEP_INIT(), STEP_SUB() and STEP_FINALIZE() are used in order to proceed the codes appropriately. A schematic representation of this strong coupling scheme was given in Figure 3.2b.

Listing A.2: Time Loops of *MainMPI.py*

```
# -------
# stage 5: time loop STRONG COUPLING

# Using strong coupling scheme:
if cpl.VD['lSTRONG'][0] == True:
 t = -1

 while cpl.stop_elli == False and \
       cpl.stop_hawc == False:

 # ---------- OUTER ITERATION ------------
  t = t + 1
  cpl.dataprocessing(t)

  if HAWC:
   if rank == 0:
    cpl_hawc.step_input(t)
    hawc.step_init()
    cpl_hawc.step_output()
   cpl_hawc.step_broadcast_val(rank,comm)

  if EL3D:
   cpl_el3d.step_input(t)
   el3d.step_init()
   cpl_el3d.step_output()

  # ---------- SUBITERATION -------------------------
  sub_t = 0
  cpl.stop_sub_hawc = False
  cpl.stop_sub_elli = False
  while cpl.stop_sub_elli == False and \
        cpl.stop_sub_hawc == False:

   sub_t = sub_t + 1
   cpl.sub_dataprocessing(sub_t)

   if EL3D:
    cpl_el3d.step_input(t)
    el3d.step_sub()
    cpl_el3d.step_output()
    cpl.stop_sub_elli=bool(cpl.VD['lEL3D_SUBEND'][0])

   if HAWC:
    if rank == 0:
     cpl_hawc.step_input(t)
     hawc.step_sub()
     cpl_hawc.step_output()
    cpl_hawc.step_broadcast_val(rank,comm)
    cpl.stop_sub_hawc=bool(cpl.VD['lHAWC_SUBEND'][0])
  # ---------------------------------------------

  if EL3D:
   cpl_el3d.step_input(t)
   el3d.step_finalize()
   cpl_el3d.step_output()
   cpl.stop_elli = bool(cpl.VD['lEL3D_END'][0])

  if HAWC:
   if rank == 0:
    cpl_hawc.step_input(t)
    hawc.step_finalize()
    cpl_hawc.step_output()
   cpl_hawc.step_broadcast_val(rank,comm)
   cpl.stop_hawc = bool(cpl.VD['lHAWC_END'][0])
```

```
# -------
# stage 5: time loop LOOSE COUPLING

# Using loose coupling scheme:
if cpl.VD['lLOOSE'][0] == True:
 t = -1

 while cpl.stop_elli == False and \
       cpl.stop_hawc == False:

 # ---------- OUTER ITERATION ------------
  t = t + 1
  cpl.dataprocessing(t)

  if STRC:
   if rank == 0:
    cpl_strc.step_input()
    strc.step()
    cpl_strc.step_output()
   cpl_strc.step_broadcast_val(rank,comm)

  if EL2D:
   cpl_el2d.step_input()
   el2d.step()
   cpl_el2d.step_output()
   cpl.stop_elli = \
                bool(cpl.VD['lEL2D_END'][0])

  if FCTR:
   if rank == 0:
    cpl_fctr.step_input()
    fctr.step()
    cpl_fctr.step_output()
   cpl_fctr.step_broadcast_val(rank,comm)

  if HAWC:
   if rank == 0:
    cpl_hawc.step_input(t)
    hawc.step_predict()
    cpl_hawc.step_output()
   cpl_hawc.step_broadcast_val(rank,comm)

  if EL3D:
   cpl_el3d.step_input(t)
   el3d.step()
   cpl_el3d.step_output()
   cpl.stop_elli = \
                bool(cpl.VD['lEL3D_END'][0])

  if HAWC:
   if rank == 0:
    cpl_hawc.step_input(t)
    hawc.step_correct()
    cpl_hawc.step_output()
   cpl_hawc.step_broadcast_val(rank,comm)
   cpl.stop_hawc = \
                bool(cpl.VD['lHAWC_END'][0])
```

The supplemental functions in the file *cpl.py*

Some of the supplemental functions gathered in the file *cpl.py* are shown in Listing A.3. They are all called from the main script *MainMPI.py*. In the function *select_models()* it is possible to select the models which should participate in the coupling. In the function *read_python_input()* the information of the input file *varlist.dat* is read in and processed. Apart from that some additional variables are set which e.g. control the start-up phase of the aero-elastic computations, select the preferred coupling scheme and define the desired printout options. Most of the remaining functions in *cpl.py* are then used to subdivide the respective functions into the model specific functions found e.g. in the file *cpl_el3d.py*. The abridgment of Listing A.3 concludes with the function *dataprocessing()* which can be used to coordinate a synchronized variable printout among the participating models.

Listing A.3: Abridgment of *cpl.py*

```
import cpl_hawc
import cpl_strc
import cpl_fctr
import cpl_el2d
import cpl_el3d

def select_models():
#-------------------------------------------
#    selecting which models are connected
#-------------------------------------------
    global HAWC, STRC, FCTR, EL2D, EL3D
    HAWC = True
    STRC = False
    FCTR = False
    EL2D = False
    EL3D = True
    return HAWC, STRC, FCTR, EL2D, EL3D


def read_python_input():
#-------------------------------------------
#    - read in varlist.dat
#    - create multidim. variable list 'VarList'
#    - create multidim. variable dictionary 'VD'
#    - set some additional variables
#-------------------------------------------
    global VarList,VD,HAWC_ROTI,EL3D_ROTI,
           print_start,print_intv

    ...

#    set some additional variables

    if HAWC and EL3D:
     # Initial rotations of Hawc and Ellipsys3D
     # before coupling starts
     HAWC_ROTI = 25
     EL3D_ROTI = 16.0
     # -> Coupling will automatically start with
     #     the same rotor position!

     # Set coupling scheme:
     VD['lSTRONG'][0] = False
     VD['lLOOSE'][0]  = True
     VD['lEnScaleFS'][0] = False

     # Output variables
     VD['lPRINTSCREEN'][0]  = False
     VD['lPRINTSEQ'][0]  = False
     print_start = 10
     print_intv  = 500
     VD['printblade'][0] = 0

    ....

def create_tailored_lists():
    if STRC: cpl_strc.create_tailored_lists()
    if FCTR: cpl_fctr.create_tailored_lists()
    if HAWC: cpl_hawc.create_tailored_lists()
    if EL2D: cpl_el2d.create_tailored_lists()
    if EL3D: cpl_el3d.create_tailored_lists()
```

```
def prepare_init_output(rank):
    if rank == 0:
     if HAWC: cpl_hawc.prepare_init_output()
     if STRC: cpl_strc.prepare_init_output()
     if FCTR: cpl_fctr.prepare_init_output()
    if EL2D: cpl_el2d.prepare_init_output()
    if EL3D: cpl_el3d.prepare_init_output()


def init_output(rank):
    if rank == 0:
     if HAWC: cpl_hawc.init_output()
     if STRC: cpl_strc.init_output()
     if FCTR: cpl_fctr.init_output()
    if EL2D: cpl_el2d.init_output()
    if EL3D: cpl_el3d.init_output()


def init_broadcast(rank,comm):
    if HAWC: cpl_hawc.init_broadcast(rank,comm)
    if STRC: cpl_strc.init_broadcast(rank,comm)
    if FCTR: cpl_fctr.init_broadcast(rank,comm)


def prepare_syn1_input(rank):
    if rank == 0:
     if HAWC: cpl_hawc.prepare_syn1_input()
     if STRC: cpl_strc.prepare_syn1_input()
     if FCTR: cpl_fctr.prepare_syn1_input()
    if EL2D: cpl_el2d.prepare_syn1_input()
    if EL3D: cpl_el3d.prepare_syn1_input()


def syn1_input(rank):
    if rank == 0:
     if HAWC: cpl_hawc.syn1_input()
     if STRC: cpl_strc.syn1_input()
     if FCTR: cpl_fctr.syn1_input()
    if EL2D: cpl_el2d.syn1_input()
    if EL3D: cpl_el3d.syn1_input()


def syn1_broadcast(rank,comm):
    if HAWC: cpl_hawc.syn1_broadcast(rank,comm)
    if STRC: cpl_strc.syn1_broadcast(rank,comm)
    if FCTR: cpl_fctr.syn1_broadcast(rank,comm)
...


def dataprocessing(t):
    VD['pyth_step'][0] = t

    VD['lPRINTNOW'][0] = False
    if t>print_start and t%print_intv == 0:
     print 'CPL.PY:_lPRINTNOW_TRUE_at_'\
           'PYTH_iteration_t:',t
     VD['lPRINTNOW'][0] = True
```

The model specific functions in the file *cpl_el3d.py*

The model specific functions of EllipSys3D are gathered in the file *cpl_el3d.py*. They are either called from the main script *MainMPI.py* or from the supplemental script *cpl.py*. The abridgment of Listing A.4 commences with the function *cpl_mode()* which sets the coupling mode of EllipSys3D. The information about the coupling mode and thus the information about which models are coupled with EllipSys3D is used inside the fluid solver in order to activate the appropriate coupling related functions. The function *cpl_mode()* also takes into consideration whether another model is indeed connected to EllipSys3D or whether it only pretends to be connected using a fully prescribed data input instead. It is also possible to run a model in a partly prescribed mode where only some of the transferred variables are prescribed. The function *create_tailored_lists()* is used to sort the data flow information of the input file *varlist.dat* into the so-called tailored lists. For each supported variable type a separate tailored list is created

containing all variables that have to be read in or read out. As an example for an actual transfer of input and output variables sent to and received from EllipSys3D the abridgment of Listing A.4 contains the function *syn1_input()*. The function loops over each item of the respective tailored lists in order to successively sent the actual variable value to EllipSys3D by finally employing the data flow functions introduced in Section 4.4.2. As a preparatory step the function *prepare_syn1_input()* is called in order to allocate the respective input variables within EllipSys3D. The function *reduce_transients()* is used to establish a smooth start-up of the aero-elastic simulations.

Listing A.4: Abridgment of *cpl_el3d.py*

```
def cpl_mode(HAWC,STRC):
    from cpl import VD

    global Instead_EL3D_Fully_Presc,\
           Instead_EL3D_Partly_Presc,\
           Instead_HAWC_Fully_Presc,\
           Instead_HAWC_Partly_Presc

    # select if EllipSys should run with fully
    # prescribed/partly prescribed input
    Instead_HAWC_Fully_Presc  = False
    Instead_HAWC_Partly_Presc = False
    Instead_STRC_Fully_Presc  = False
    Instead_STRC_Partly_Presc = False

    # assign dimensions
    VD['lHAWC'][1]        = 1
    VD['lSTRC'][1]        = 1

    # run EllipSys in HAWC-Mode, only if either
    # HAWC=True or if Fully_Presc=True
    # or if Partly_Presc=True
    if HAWC or Instead_HAWC_Fully_Presc \
            or Instead_HAWC_Partly_Presc:
     VD['lHAWC'][0]       = True
    else:
     VD['lHAWC'][0]       = False

    # run EllipSys in STRC-Mode, only if either
    # STRC=True or if Fully_Presc=True
    # or if Partly_Presc=True
    if STRC or Instead_STRC_Fully_Presc \
            or Instead_STRC_Partly_Presc:
     VD['lSTRC'][0]       = True
    else:
     VD['lSTRC'][0]       = False


def create_tailored_lists():

#-----------------------------------------
#   - split up 'VarList' and create tailored
#     variable lists
#-----------------------------------------

    from cpl import VarList
    import numpy as np

    # make all tailored lists globally available
    global BOOL_EL3D_PYTH,REAL_EL3D_PYTH,\
           CHAR_EL3D_PYTH,ITGR_EL3D_PYTH,\
           IT1D_EL3D_PYTH,RL1D_EL3D_PYTH,\
           RL2D_EL3D_PYTH,RL3D_EL3D_PYTH,\
           BOOL_PYTH_EL3D,REAL_PYTH_EL3D,\
           CHAR_PYTH_EL3D,ITGR_PYTH_EL3D,\
           IT1D_PYTH_EL3D,RL1D_PYTH_EL3D,\
           RL2D_PYTH_EL3D,RL3D_PYTH_EL3D
    ...


def prepare_syn1_input():

#-----------------------------------------
#   allocate the input variables of EllipSys at
#    SYN1
#-----------------------------------------

    from cpl import VD
    import ellipsys3D_MPI as el3d

    # prescribe dimensions for fully prescribed
    #       or
    # partly prescribed mode
    syn1_prescribed_input_dim()

    for item in IT1D_PYTH_EL3D:
     if item[5] == 'SYN1':
      el3d.prep_input_it1d(len(item[0]),item[0],\
                           VD[item[0]][1])
    for item in RL1D_PYTH_EL3D:
     if item[5] == 'SYN1':
      el3d.prep_input_rl1d(len(item[0]),item[0],\
                           VD[item[0]][1])
    for item in RL2D_PYTH_EL3D:
     if item[5] == 'SYN1':
      vardim = VD[item[0]][1]
      el3d.prep_input_rl2d(len(item[0]),item[0],\
                           vardim[0],vardim[1])
    for item in RL3D_PYTH_EL3D:
     if item[5] == 'SYN1':
      vardim = VD[item[0]][1]
      el3d.prep_input_rl3d(len(item[0]),item[0],\
                           vardim[0],vardim[1],vardim[2])


def syn1_input():

#-----------------------------------------
#   send the input variables of EllipSys at SYN1
#-----------------------------------------

    from cpl import VD
    import ellipsys3D_MPI as el3d

    # prescribe variables for fully prescribed
    # or partly prescribed mode
    syn1_prescribed_input_val()

    # send to EllipSys
    for item in BOOL_PYTH_EL3D:
     if item[5] == 'SYN1':
      el3d.input_bool(len(item[0]),item[0],\
                      VD[item[0]][0])
    for item in REAL_PYTH_EL3D:
     if item[5] == 'SYN1':
      el3d.input_real(len(item[0]),item[0],\
                      VD[item[0]][0])
    for item in CHAR_PYTH_EL3D:
     if item[5] == 'SYN1':
      el3d.input_char(len(item[0]),item[0],\
                      VD[item[0]][0])
    for item in ITGR_PYTH_EL3D:
     if item[5] == 'SYN1':
      el3d.input_itgr(len(item[0]),item[0],\
                      VD[item[0]][0])
    for item in IT1D_PYTH_EL3D:
     if item[5] == 'SYN1':
      el3d.input_it1d(len(item[0]),item[0],\
                      VD[item[0]][1],VD[item[0]][0])
    for item in RL1D_PYTH_EL3D:
     if item[5] == 'SYN1':
      el3d.input_rl1d(len(item[0]),item[0],\
                      VD[item[0]][1],VD[item[0]][0])
    for item in RL2D_PYTH_EL3D:
     if item[5] == 'SYN1':
      vardim = VD[item[0]][1]
      el3d.input_rl2d(len(item[0]),item[0],\
                      vardim[0],vardim[1],VD[item[0]][0])
    for item in RL3D_PYTH_EL3D:
     if item[5] == 'SYN1':
      vardim = VD[item[0]][1]
      el3d.input_rl3d(len(item[0]),item[0],\
                      vardim[0],vardim[1],vardim[2],\
                      VD[item[0]][0])


def reduce_transients(rank):
    ...
```

A.2 Synchronized Printout of Variables

The solvers that participate in the developed Python coupling framework of the present work have been equipped with particular printout routines which write out intermediate and final results into certain output files. In order to synchronize this printout and assure that the results of a coupled simulation are written out within the same time steps the printout procedure is managed and triggered by the superordinate Python coupling framework. In the file *cpl.py* several printout variables can be set in order to e.g. decide in which sequence the synchronized printout should be done and on which turbine blades the printout values should be focused.

A.3 Reduced Transients at Start-Up

Before the solvers enter the time loop of the coupled simulation it is possible to define a certain start-up procedure in order to reduce the initial transients of the aero-elastic simulation and in order to speed up the overall computation time. The procedure is called in Stage 4 of the main script *MainMPI.py*.

The particular start-up procedure defined for the aero-elastic simulation using HAWC2 and EllipSys3D can be found in Appendix B.1.

A.4 Coupling Conventions

When former stand-alone solvers are joined together it is necessary to respect some basic conventions in order to handle or avoid ambivalent data input.

The conventions particularly needed for a successful coupling between HAWC2 and EllipSys3D are gathered in Appendix B.2.

A.5 File Structure of the Participating Solvers

The explanations of Appendix A were so far exclusively dealing with the functionality that was implemented in the Python part of the coupling framework. However, in order to employ the presented Python coupling framework it is also necessary to prepare the participating solvers in a particular way. Each of the connected solvers is therefore equipped with three additional files written

in the programming language of the respective code. In case the solver is written in Fortran 90 those three files are named *py_interface.f90*, *cpl_eqns.f90* and *cpl_modules.f90*. The file *py_interface.f90* contains the definitions of the standardized interface functions presented in Section 4.4 and is used by *F2Py* during the wrapping process. The file *cpl_eqns.f90* contains the coupling related functions mentioned in Section 4.5 and the file *cpl_modules.f90* contains all additional variables needed by those coupling related functions.

Appendix B

About the Coupling between HAWC2 and EllipSys3D

B.1 Start-Up Procedure

Starting up an aero-elastic computation usually results in a relatively long time period where the aerodynamic forces and the structural deflections are not yet in equilibrium. In terms of a pure HAWC2 simulation where the traditional BEM method is used in order to calculate the aerodynamic forces this start-up phase can take up to 25 rotor rotations. Additionally, the CFD solver EllipSys3D also requires a certain time period until the flow field is settled and the wake is fully developed. In terms of a pure EllipSys3D computation where a stiff rotor structure is used this can take up to 20 rotor rotations.

In EllipSys3D the problem of this long and computationally expensive start-up procedure can be reduced by the possibility of stopping and then restarting the simulation at a certain time instant. In this way it is not necessary to rerun the same start-up phase for similar simulations. Unfortunately this feature of restarting a simulation at a certain time instant is not available in HAWC2 and it is thus not possible to stop and restart an aero-elastic computation.

However, in order to optimize the computational costs of the start-up phase the following procedure could be implemented:

1. Run the traditional HAWC2 code with the computationally cheap BEM aerodynamics until the transients are dampened out and a certain equilibrium between structural deformations and aerodynamic forces is found.

2. Run a traditional EllipSys3D computation with a stiff rotor structure using the structural deformations of the equilibrium state of Step 1. A restart file can be saved at the end of this step. In this way it is not necessary to rerun this step for similar simulations.

3. Start the coupled FSI simulation by applying the settled aerodynamic forces of Step 2 on the settled structural deformations of Step 1.

Using the computationally cheap BEM method in Step 1 reduces the overall simulation time significantly. While Step 1 and Step 3 are part of the aero-elastic computation which can't be restarted, at least Step 2 can be omitted for similar simulations by using a restart file instead. This reduces the computational time of the start-up phase decisively.

The algorithm takes care that both codes have exactly the same structural position when the codes are clutched together at Step 3. The only additional input needed for this start-up procedure is the number of rotor rotations HAWC2 should simulate during Step 1 and the number of rotor rotations EllipSys3D should simulate during Step 2. The respective variables can be found in Listing A.3.

B.2 Coupling Conventions

Some few conventions have to be made in order to harmonize the coupled execution of HAWC2 and EllipSys3D:

- The time step size of the coupled simulation is the one chosen in the input file of EllipSys3D. The time step size defined in the input file of HAWC2 is overwritten.

- In the input files of both models the initial position of blade 1 has to be defined in a way that the blade is pointing upwards, i.e. it has to point along the positive $y - axis$ in the $GLCS_F$ of EllipSys3D and along the negative $y - axis$ in the $SHCS_S^{Ref}$ of HAWC2.

- The models have to be built up by using the typical axis conventions of the stand-alone codes.

B.3 Axis Transformation at Domain Interface

At the interface between HAWC2 and EllipSys3D the force and deflection transfer has to consider the following changes in the axis orientation.

- $GRCS_S \mapsto GLCS_F$:
  $x_S^{GRCS} \mapsto x_F^{GRCS}, \quad y_S^{GRCS} \mapsto z_F^{GRCS}, \quad z_S^{GRCS} \mapsto -y_F^{GRCS}$ [1]
- $SHCS_S \mapsto SHCS_F$:
  $x_S^{SHCS} \mapsto x_F^{SHCS}, \quad y_S^{SHCS} \mapsto -y_F^{SHCS}, \quad z_S^{SHCS} \mapsto -z_F^{SHCS}$
- $HUCS_S \mapsto HUCS_F$:
  $x_S^{HUCS} \mapsto -x_F^{HUCS}, \quad y_S^{HUCS} \mapsto z_F^{HUCS}, \quad z_S^{HUCS} \mapsto y_F^{HUCS}$
- $BLCS_S \mapsto BLCS_F$:
  $x_S^{BLCS} \mapsto -x_F^{BLCS}, \quad y_S^{BLCS} \mapsto z_F^{BLCS}, \quad z_S^{BLCS} \mapsto y_F^{BLCS}$
- $BSCS_S \mapsto BSCS_F$:
  $x_S^{BSCS} \mapsto -x_F^{BSCS}, \quad y_S^{BSCS} \mapsto z_F^{BSCS}, \quad z_S^{BSCS} \mapsto y_F^{BSCS}$

B.4 Outline of an Energy Conservative Load Transfer

As mentioned in Section 3.2.7 and Section 5.3.1 an energy conservative load and motion transfer between the highly non-matching meshes of HAWC2 and EllipSys3D should be feasible by following the approach suggested in Brown [72]. The following draft will give an idea about how this approach can be implemented in the FSI coupling between HAWC2 and EllipSys3D.

In Section 3.2.7 it was explained that an energy conservative load and motion transfer can be achieved if the displacements of the fluid mesh are expressed via the interpolation functions of the structure gathered in $\boldsymbol{\eta}(\mathbf{x_u})$, and it is shown in the following how a similar expression can be derived for the considered FSI coupling between HAWC2 and EllipSys3D. For simplicity, the derivation only focuses on the load and motion transfer connected to one particular beam element E as illustrated in Figure B.1. In HAWC2 each of the two end nodes $\mathbf{q}_{E,1}$ and $\mathbf{q}_{E,2}$ has three translational and three rotational degrees of freedom resulting in a total amount of twelve degrees of freedom per beam element. Gathering the

[1] The transformation includes a fixed translatory offset

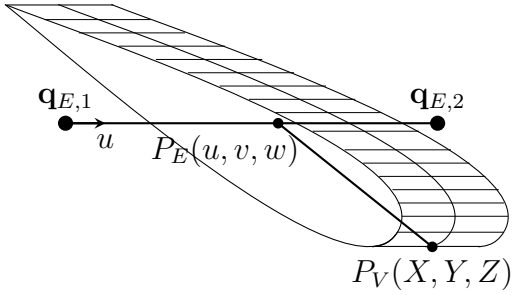


Figure B.1: Extrapolation of Mesh Deformation using Brown's Approach

twelve respective deflections in the vector $\mathbf{q}_E$ and denoting the interpolation or shape functions of the considered beam element with $\boldsymbol{\eta}_E(u)$ where u is the coordinate along the beam or blade axis, the continuously defined deflections $\mathbf{u}_E(u)$ of the beam are given by

$$\mathbf{u}_E(u) = \boldsymbol{\eta}_E(u) \cdot \mathbf{q}_E \tag{B.1}$$

Dividing the six components of the array $\mathbf{u}_E(u)$ into the three translational deflections $\mathbf{u}_E^t(u)$ and the three rotational deflections $\mathbf{u}_E^r(u)$ the above equation can be split into

$$\mathbf{u}_E^t(u) = \boldsymbol{\eta}_E^t(u) \cdot \mathbf{q}_E \tag{B.2}$$

$$\mathbf{u}_E^r(u) = \boldsymbol{\eta}_E^r(u) \cdot \mathbf{q}_E \tag{B.3}$$

The continuously defined deflections of Equation B.2 and Equation B.3 can now be extrapolated to the three-dimensional CFD mesh that encloses the beam. Assuming that the cross sections of the blade do not change their shape during the deformation a stiff connection can be drawn from every CFD surface mesh point to the deflected beam element. An exemplary connection is illustrated in Figure B.1 where the mesh vertex P_V is linked to the respective projected point P_E on the beam. By employing the assumption of having rigid blade sections the translational motion $\mathbf{x}_V^t$ of any vertex V located on the CFD surface mesh can be expressed with the formula

$$\mathbf{x}_V^t(u) = \mathbf{u}_E^t(u) - \begin{bmatrix} 0 & w-Z & Y-v \\ Z-w & 0 & u-X \\ v-Y & X-u & 0 \end{bmatrix} \cdot \mathbf{u}_E^r(u) \tag{B.4}$$

using *Poisson's form of the cross product* as given in Brown [72]. Here, X, Y, Z

are the coordinates of vertex V and u, v, w are the coordinates of the respective projection point on the deflected beam element E. In the considered case of Figure B.1 the coordinate u is equal to the coordinate X and the projected coordinate u is thus directly known for each CFD mesh vertex V.

In order to accomplish the mesh motion in EllipSys3D it is sufficient to know the three translational coordinates $\mathbf{x}_V^t$, however, it is also possible to transform the rotational components from beam E to CFD mesh vertex V by using the simple relation

$$\mathbf{x}_V^r(u) = \mathbf{u}_E^r(u) \tag{B.5}$$

Equation B.4 can be rewritten with

$$\mathbf{x}_V^t(u) = \left(\boldsymbol{\eta}_E^t(u) - \begin{bmatrix} 0 & w-Z & Y-v \\ Z-w & 0 & u-X \\ v-Y & X-u & 0 \end{bmatrix} \cdot \boldsymbol{\eta}_E^r(u) \right) \cdot \mathbf{q}_E = \mathbf{N^t}(u) \cdot \mathbf{q}_E \tag{B.6}$$

where $\mathbf{N^t}(u)$ is the extrapolation function that extrapolates the beam deflections to the translational motion of the surrounding CFD mesh. In the same way, Equation B.5 can be rewritten with

$$\mathbf{x}_V^r(u) = \boldsymbol{\eta}_E^r(u) \cdot \mathbf{q}_E = \mathbf{N^r}(u) \cdot \mathbf{q}_E \tag{B.7}$$

where $\mathbf{N^r}(u)$ is the extrapolation function for the rotations. By gathering the translational and rotational components in one single equation

$$\begin{bmatrix} \mathbf{x}_V^t(u) \\ \mathbf{x}_V^r(u) \end{bmatrix} = \begin{bmatrix} \mathbf{N^t}(u) \\ \mathbf{N^r}(u) \end{bmatrix} \cdot \mathbf{q}_E \tag{B.8}$$

$$\mathbf{x}_V(u) = \mathbf{N}(u) \cdot \mathbf{q}_E \tag{B.9}$$

an expression is obtained that is comparable to the general formulation of Equation 3.13 using the extrapolation function $\mathbf{N}(u)$ instead of the interpolation function $\boldsymbol{\eta}(\mathbf{x_u})$. The transpose of the extrapolation function $\mathbf{N}(u)$ can finally be used to transform the fluid forces $\mathbf{F}_{F,V}$ of vertex V to the respective nodal loads $\mathbf{F}_{S,E}$ of the beam element E

$$\mathbf{F}_{S,E} = \mathbf{N}(u)^T \cdot \mathbf{F}_{F,V} \tag{B.10}$$

which is an expression comparable to the general formulation of Equation 3.15.

Remark:
The transposed extrapolation function $\mathbf{N}(u)^T$ of the present example is a $12x6$ matrix and distributes the six force components of mesh vertex V to the twelve degrees of freedom of beam element E. The contributions of all other vertices V that are connected to beam element E are then simply added up on top.

Bibliography

[1] T.J. Larsen. How 2 HAWC2 the User's Manual. Technical Report R-1597(EN), Risø National Laboratory for Sustainable Energy, Denmark, 2009.

[2] J.A. Michelsen. *Basis3D / A Platform for Development of Multiblock PDE Solvers*. Technical University of Denmark Department of Fluid Mechanics, Lyngby, 1992.

[3] J.A. Michelsen. *Block Structured Multigrid Solution of 2D and 3D Elliptic PDE's*. Technical University of Denmark Department of Fluid Mechanics, Lyngby, 1994.

[4] N. N. Sørensen. General Purpose Flow Solver Applied to Flow over Hills. Technical Report Risø-R-827(EN), Risø National Laboratory, Roskilde, 1995.

[5] Geuzaine, Brown, Harris, and Farhat. Aeroelastic Dynamic Analysis of a Full F-16 Configuration for Various Flight Conditions. *AIAA Journal*, 41(3):363–371, 2003.

[6] C. Farhat, M. Lesoinne, and P. LeTallec. Load and Motion Transfer Algorithms for Fluid/Structure Interaction Problems with Non-Matching Discrete Interfaces: Momentum and Energy Conservation, Optimal Discretization and Application to Aeroelasticity. *Computer Methods in Applied Mechanics and Engineering*, 157(1-2):95–114, 1998.

[7] Serge Piperno and Charbel Farhat. Partitioned Procedures for the Transient Solution of Coupled Aeroelastic Problems - Part II: Energy Transfer Analysis and Three-Dimensional Applications. *Computer Methods in Applied Mechanics and Engineering*, 190(24-25):3147–3170, 2001.

[8] Charbel Farhat, Kristoffer G. van der Zee, and Philippe Geuzaine. Provably Second-Order Time-Accurate Loosely-Coupled Solution Algorithms for

Transient Nonlinear Computational Aeroelasticity. *Computer Methods in Applied Mechanics and Engineering*, 195(17-18):1973–2001, 2006.

[9] Daniel P Mok. *Partitionierte Lösungsansätze in der Strukturdynamik und der Fluid-Struktur-Interaktion*. Institut für Baustatik, Stuttgart, 2001.

[10] WA Wall, DP Mok, and E Ramm. Iterative Substructering Schemes for Fluid Structure Interaction. *Analysis and Simulation of Multifield Problems*, 12:349–381, 2003.

[11] P. Causin, J.F. Gerbeau, and F. Nobile. Added-Mass Effect in the Design of Partitioned Algorithms for Fluid-Structure Problems. *Computer Methods in Applied Mechanics and Engineering*, 194(42-44):4506–4527, 2005.

[12] Christiane Förster, Wolfgang A. Wall, and Ekkehard Ramm. Artificial Added Mass Instabilities in Sequential Staggered Coupling of Nonlinear Structures and Incompressible Viscous Flows. *Computer Methods in Applied Mechanics and Engineering*, 196(7):1278–1293, 2007.

[13] Hao Song, Longbin Tao, and Subrata Chakrabarti. Modelling of Water Wave Interaction with Multiple Cylinders of Arbitrary Shape. *Journal of Computational Physics*, 229(5):1498–1513, 2010.

[14] Carlotta Costa, Claudio Borri, Olivier Flamand, and Gérard Grillaud. Time-Domain Buffeting Simulations for Wind-Bridge Interaction. *Journal of Wind Engineering & Industrial Aerodynamics*, 95(9-11):991–1006, 2007.

[15] Tayfun E. Tezduyar, Kenji Takizawa, Creighton Moorman, Samuel Wright, and Jason Christopher. Space-Time Finite Element Computation of Complex Fluid-Structure Interactions. *International Journal for Numerical Methods in Fluids*, 64(1012):1201–1218, 2010.

[16] P.-O. Marklund and L. Nilsson. Simulation of Airbag Inflation Processes using a Coupled Fluid Structure Approach. *Computational Mechanics*, 29(4-5):289–297, 2002.

[17] A. S. Veletsos and Y. Tang. Soil-Structure Interaction Effects for Laterally Excited Liquid Storage Tanks. *Earthquake Engineering & Structural Dynamics*, 19(4):473–496, May 1990.

[18] M.O.L. Hansen, J.N. Sørensen, S. Voutsinas, N. Sørensen, and H.Aa. Madsen. State of the Art in Wind Turbine Aerodynamics and Aeroelasticity. *Progress in Aerospace Sciences*, 42(4):285–330, 2006.

[19] Pinting Zhang and Shuhong Huang. Review of Aeroelasticity for Wind Turbine: Current Status, Research Focus and Future Perspectives. *Frontiers in Energy*, 5(4):419–434, 2011.

[20] J.M. Jonkman and M.L. Buhl Jr. FAST User's Guide. Technical Report NREL/EL-500-29798, National Renewable Energy Laboratory, Golden, Colorado, 2005.

[21] S. Øye. FLEX4 Simulation of Wind Turbine Dynamics. In *Proceedings of the 28th IEA Meeting of Experts Concerning State of the Art of Aeroelastic Codes for Wind Turbine Calculations*, pages 129–135, Lyngby, Denmark, 1996. Pedersen.

[22] E.A. Bossanyi. GH Bladed Theory Manual. Technical Report, GH & Partners Ltd, 2003.

[23] C. Lindenberg. Phatas Release "NOV-2003" and "APR-2005" User's Manual. Technical Report ECN-I–05-005, Energy Research Center of the Netherlands, ECN, 2005.

[24] V.A. Riziotis and S.G. Voutsinas. GAST: A General Aerodynamic and Structural Prediction Tool for Wind Turbines. In *Proceedings of the EWEC*, Dublin, Ireland, 1997.

[25] Bjarne Skovmose Kallesøe and Peter Bjerring. Global Blade Deflections Effect on Local Airfoil Deformation and Performance. *Research in Aeroelasticity EFP-2007-II*, pages 122–133, 2009.

[26] J.P. Blasques. User's Manual for BECAS. A Cross Section Analysis Tool for Anisotropic and Inhomogeneous Beam Sections of Arbitrary Geometry. Technical Report Risø-R-1785, Risø National Laboratory for Sustainable Energy, Denmark, 2012.

[27] Niels N. Sørensen and M.O.L. Hansen. Rotor Performance Predictions using a Navier-Stokes Method. In *A Collection of the 1998 ASME Wind Energy Symposium Technical Papers*, pages 52–59, 1998.

[28] Xu Guanpeng and L.N. Sankar. Computational Study of Horizontal Axis Wind Turbines. *Transactions of the ASME. Journal of Solar Energy Engineering*, 122(1):35–39, 2000.

[29] E. P. N. Duque, C. P. van Dam, and S. C. Hughes. Navier-Stokes Simulations of the NREL Combined Experiment Phase II Rotor. In *ASME wind energy symposium*, Reno, 1999.

[30] L. Fingersh, D. Simms, M. Hand, D. Jager, J. Cotrell, and M. Robinson. Wind Tunnel Testing of NREL's Unsteady Aerodynamics Experiment. *American Institute of Aeronautics and Astronautics*, 1(20/67):194–200, 2001.

[31] N.N. Sørensen, J.A. Michelsen, and S. Schreek. Navier-Stokes Predictions of the NREL Phase VI Rotor in the NASA Ames 80-by-120 Wind Tunnel. In *ASME 2002 Wind Energy Symposium, WIND2002*, pages 94–105, 2002.

[32] Niels N. Sørensen. UPWIND, Aerodynamics and Aero-Elasticity Rotor Aerodynamics in Atmospheric Shear Flow. In *EWEC*, 2007.

[33] J. Johansen, Niels N. Sørensen, J.A. Michelsen, and S. Schreck. Detached-Eddy Simulation of Flow around the NREL Phase-VI Rotor (Poster). In *Proceedings CD-ROM. CD 2*, Madrid, Spain, 2003.

[34] Oliver Fleig and Chuichi Arakawa. Numerical Simulation of Wind Turbine Tip Noise. In *Collection of ASME Wind Energy Symposium Technical Papers*, pages 587–597, 2004.

[35] Frederik Zahle, Niels N. Sørensen, and Jeppe Johansen. Wind Turbine Rotor-Tower Interaction using an Incompressible Overset Grid Method. *Wind Energy*, 12(6):594–619, 2009.

[36] A. Bechmann, N. N. Sørensen, and F. Zahle. CFD Simulations of the MEXICO Rotor. *Wind Energy*, 14(5):677–689, 2011.

[37] H. Snel, J.G. Schepers, and N.B. Siccama. Mexico Project: The Database and Results of Data Processing and Interpretation. In *47th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, pages 2009–1217, 2009.

[38] Ming-Chen Hsu and Yuri Bazilevs. Fluid-Structure Interaction Modeling of Wind Turbines: Simulating the Full Machine. *Computational Mechanics*, 50(6):821–833, 2012.

[39] Yuri Bazilevs, Ming-Chen Hsu, Kenji Takizawa, and Tayfun E. Tezduyar. ALE-VMS and ST-VMS Methods for Computer Modeling of Wind-Turbine Rotor Aerodynamics and Fluid-Structure Interaction. *Mathematical Models and Methods in Applied Sciences*, 22, 2012.

[40] T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs. Isogeometric Analysis: CAD, Finite Elements, NURBS, Exact Geometry and Mesh Refinement. *Computer Methods in Applied Mechanics and Engineering*, 194(39-41):4135–4195, 2005.

[41] G Schepers. *Engineering models in wind energy aerodynamics development, implementation and analysis using dedicated aerodynamic measurements: proefschrift*. Techische Universiteit Delft, Delft, 2012.

[42] M.H. Hansen, Mac Gaunaa, and Helge Aagaard Madsen. A Beddoes-Leishman Type Dynamic Stall Model in State-Space and Indicial Formulations. Technical Report Risø-R-1354, Risø National Laboratory for Sustainable Energy, Denmark, 2004.

[43] T. Theodorsen. General Theory of Aerodynamic Instability and Mechanism of Flutter. Technical Report 496, National Advisory Committee for Aeronautics (United States Avisory Committee for Aeronautics), 1935.

[44] Raymond L Bisplinghoff, Holt Ashley, and Robert L Halfman. *Aeroelasticity*. Dover ; Constable, New York; London, 1996.

[45] M. Gaunaa. Unsteady 2D Potential-Flow Forces on a Thin Variable Geometry Airfoil undergoing Arbitrary Motion. Technical Report Risø-R-1478, Risø National Laboratory for Sustainable Energy, Denmark, 2004.

[46] L. Bergami and M. Gaunaa. ATEFlap Aerodynamic Model, A Dynamic Stall Model including the Effects of Trailing Edge Flap Deflection. Technical Report Risø-R-1792(EN), Risø National Laboratory for Sustainable Energy, Denmark, 2012.

[47] Leonardo Bergami, Mac Gaunaa, and Joachim Heinz. Indicial Lift Response Function: An Empirical Relation for Finite-Thickness Airfoils, And Effects on Aeroelastic Simulations. *Wind Energy*, June 2012.

[48] H. Snel and J.G. Schepers. Joint Investigation of Dynamic Inflow Effects and Implementation of an Engineering Method. Technical Report ECN-C-94-107, ECN, Netherlands Energy Research Foundation, 1995.

[49] H. Aa. Madsen, V. Riziotis, F. Zahle, M.O.L. Hansen, H. Snel, F. Grasso, T.J. Larsen, E. Politis, and F. Rasmussen. Blade Element Momentum Modeling of Inflow with Shear in Comparison with Advanced Model Results. *Wind Energy*, 15(1):63–81, 2012.

[50] G.C. Larsen, H. Aagaard Madsen, T.J. Larsen, and N. Troldborg. Wake Modeling and Simulation. Technical Report Risø-R-1653, Risø National Laboratory for Sustainable Energy, Denmark, 2008.

[51] Peter Bjørn Andersen, Helge Aagaard Madsen, and Mac Gaunaa. A Near Wake Model for Deformable Trailing Edge Flaps Implemented in the Multi-Body Aero-Servo-Elastic Code HAWC2. In *EWEC 2010 Proceedings Online*, 2010.

[52] Srinivas Guntur, Christian Bak, and Niels N. Sørensen. Analysis of 3D Stall Models for Wind Turbine Blades Using Data from the MEXICO Experiment. *Proceedings of the 13th International Conference on Wind Engineering*, 2012.

[53] Thomas Buhl, Mac Gaunaa, and Christian Bak. Potential Load Reduction Using Airfoils with Variable Trailing Edge Geometry. *Journal of Solar Energy Engineering*, 127(4):503, 2005.

[54] Frederik Zahle, Helge Aagaard Madsen, and Niels N. Sørensen. Unsteady Navier-Stokes Simulations of a Rotor Operating in Wake. In *EWEA Proceedings*, Brussels, Belgium, 2011.

[55] Joachim Heinz, Niels N. Sørensen, and Frederik Zahle. Investigation of the Load Reduction Potential of Two Trailing Edge Flap Controls using CFD. *Wind Energy*, 14(3):449–462, April 2011.

[56] F. Bertagnolio, Niels N. Sørensen, M.H. Madsen, and M. Gaunaa. Aeroelastic Simulation of a Wind Turbine Airfoil by Coupling CFD and a Beam Element Model. In *Proceedings CD-ROM. CD 2*, Madrid, Spain, 2003.

[57] J. Thirstrup Petersen. The Aeroelastic Code HawC - Model and Comparisons. In *State of the Art of Aeroelastic Codes for Wind Turbine Calculations*, pages 129–135, Technical University of Denmark, 1996.

[58] J. Jonkman, S. Butterfield, W. Musial, and G. Scott. Definition of a 5-MW Reference Wind Turbine for Offshore System Development. Technical Report NREL/TP-500-38060, NREL, Golden, Colorado, February 2009.

[59] Passon, Kuhn, Butterfield, Jonkman, Camp, and Larsen. OC3-Benchmark Exercise of Aero-Elastic Offshore Wind Turbine Codes. *Journal of Physics: Conference Series*, 75(1), 2007.

[60] Torben J. Larsen, Helge Aa. Madsen, Gunner C. Larsen, and Kurt S. Hansen. Validation of the Dynamic Wake Meander Model for Loads and Power Production in the Egmond Aan Zee Wind Farm. *Wind Energy*, October 2012.

[61] Steen Krenk. *Non-Linear Modeling and Analysis of Solids and Structures*. Cambridge University Press, Cambridge, 2009.

[62] C. M. Rhie and W. L. Chow. Numerical Study of the Turbulent Flow past an Airfoil with Trailing Edge Separation. *AIAA Journal*, 21(11):1525–1532, November 1983.

[63] Joel H Ferziger and Milovan Peric. *Numerische Strömungsmechanik*. Springer, Berlin; Heidelberg, 2008.

[64] FR Menter. 2-Equation Eddy-Viscosity Turbulence Models for Engineering Application. *AIAA Journal*, 32(8):1598–1605, 1994.

[65] Menter. Performance of Popular Turbulence Models for Attached and Separated Adverse Pressure Gradient Flows. *AIAA Journal*, 30(8):2066–2072, 1992.

[66] Philippe Geuzaine, Celine Grandmont, and Charbel Farhat. Design and Analysis of ALE Schemes with Provable Second-Order Time-Accuracy for Inviscid and Viscous Flow Simulations. *Journal of Computational Physics*, 191(1):206–227, 2003.

[67] C Farhat. Design and Analysis of Robust ALE Time-Integrators for the Solution of Unsteady Flow Problems on Moving Grids. *Computer Methods in Applied Mechanics and Engineering*, 193:4073–4095, October 2004.

[68] W.R. Skrzypinski, M. Gaunaa, N.N. Sørensen, F. Zahle, and J. Heinz. Self-Induced Vibrations of a DU96-W-180 Airfoil in Stall (in Review Process). *Wind Energy*, 2013.

[69] W.R. Skrzypinski, M. Gaunaa, N.N. Sørensen, F. Zahle, and J. Heinz. Vortex-Induced Vibrations of a DU96-W-180 Airfoil at 90 Degrees Angle of Attack (in Review Process). *Wind Energy*, 2013.

[70] Stig Øye. Aeroelastic Modelling of Wind Turbines (Lecture Notes). Technical report, Technical University of Denmark, 2006.

[71] C. Farhat and M. Lesoinne. On the Accuracy, Stability and Performance of the Solution of Three-Dimensional Nonlinear Transient Aeroelastic Problems by Partitioned Procedures. *Collection of Technical Papers - AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, 1:629–641, 1996.

[72] S.A. Brown. Displacement Extrapolations for CFD+CSM Aeroelastic Analysis. *Collection of Technical Papers - AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, 1:291–300, 1997.

[73] A. de Boer, A.H. van Zuijlen, and H. Bijl. Comparison of Conservative and Consistent Approaches for the Coupling of Non-Matching Meshes. *Computer Methods in Applied Mechanics and Engineering*, 197(49-50):4284–4297, 2008.

[74] A. De Boer. *Computational Fluid-Structure Interaction: Spatial Coupling, Coupling Shell and Mesh Deformation.* Dissertation, Technical University Delft, Aerospace Engineering, 2008.

[75] H.J.T. Kooijman, C. Lindenburg, D. Winkelaar, and E.L. Van der Hooft. DOWEC 6MW pre-design: Aero-elastic modelling of the DOWEC 6 MW pre-design in PHATAS. Public Reports ECN-CX–01-135, Energy Research Center of the Netherlands, 2003.

[76] W. Z. Shen, Jess Michelsen, and J. N. Sørensen. An Improved Rhie-Chow Interpolation for Unsteady Flow Computations. *American Institute of Aeronautics and Astronautics, Inc.*, 39(12):2406—2409, 2001.

This dissertation is submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy in Engineering at the Technical University of Denmark. It is based on the work done during a three years Ph.D. study at the Aeroelastic Design Section of the Department of Wind Energy at Risø Campus. The work was partly funded by the ATEF project on adaptive trailing edge flaps for wind turbines sponsored by Højteknologifonden. The dissertation was submitted in February 2013 and successfully defended the 17th of June 2013.

Main Supervisor: Research Professor, Niels N. Sørensen, DTU Wind Energy.
Co-Supervisor: Senior Scientist, Frederik Zahle, DTU Wind Energy.
Co-Supervisor: Senior Researcher, John M. Hansen, DTU Wind Energy.

DTU Wind Energy
Technical University of Denmark

Risø Campus, 118
Frederiksborgvej 399
DK-4000 Roskilde
http://www.vindenergi.dtu.dk/

ISBN 978-87-92896-74-2